TEXTURED MOTION ANALYSIS

A THESIS SUBMITTED TO THE
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

KAAN ÖZTEKİN

IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2005

Approval of the Graduate School of Natural and Applied Sciences.

____

Prof. Dr. Canan ÖZGEN

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

____

Prof. Dr. İsmet ERKMEN

Head of the Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

____

Assoc. Prof. Dr. Gözde BOZDAĞI AKAR

Supervisor

Examining Committee Members

Prof. Dr. İsmet ERKMEN	(METU,EE)	__________________
Assoc. Prof. Dr. Gözde BOZDAĞI AKAR	(METU,EE)	__________________
Prof. Dr. Adnan YAZICI	(METU,CENG)	__________________
Assoc. Prof. Dr. Aydın ALATAN	(METU,EE)	__________________
Dr. Çağatay CANDAN	(METU,EE)	__________________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Kaan ÖZTEKİN

ABSTRACT

TEXTURED MOTION ANALYSIS

Öztekin, Kaan
M.S., Department of Electrical and Electronics Engineering
Supervisor: Assoc.Prof.Dr. Gözde Bozdağı Akar

December 2005, 89 pages

Textured motion - generally known as dynamic or temporal texture - is a popular research area for synthesis, segmentation and recognition. Dynamic texture is a spatially repetitive, time-varying visual pattern that forms an image sequence with certain temporal stationarity. In dynamic texture, the notion of self-similarity central to conventional image texture is extended to the spatiotemporal domain. Dynamic textures are typically videos of processes, such as waves, smoke, fire, a flag blowing in the wind, a moving escalator, or a walking crowd. Creation of synthetic frames is a key issue especially for movie screen industry to enrich their scenes from a white screen into a shining reality. In robotics world, for example an autonomous vehicle must decide what is traversable terrain (e.g. grass) and what is not (e.g. water). This problem can be addressed by classifying portions of the image into a number of categories, for instance grass, dirt, bushes or water. If these parts are identifiable, then segmentation and recognition of these textures results with an efficient path planning for the autonomous vehicle. In this thesis, we aimed to characterize these textured motions like mentioned above. We tried to implement several known techniques and compared the results.

Keywords: Dynamic Texture, Temporal Texture, Texture Motion, Texture Recognition, Texture Classification

ÖZ

DOKULU HAREKET ÇÖZÜMLEMESİ

Öztekin, Kaan
Yüksek Lisans, Elektrik-Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Doç.Dr. Gözde Bozdağı Akar

Aralık 2005, 89 sayfa

Dokulu Hareket – genellikle bilinen tanımı ile dinamik veya zamansal doku - sentezleme, bölümleme ve tanıma alanlarında popüler bir araştırma alanıdır. Dinamik doku, bir örüntü dizisini belirli zamansal durağanlıkla oluşturan, uzaysal boyutta tekrar eden, zamanla değişen bir görsel desendir. Dinamik dokuda geleneksel örüntü dokusunun merkezinde yer alan öz benzerlik fikri uzay-zaman boyutuna genişletilmiştir. Dinamik dokular, dalgalar, duman, ateş, rüzgarda sallanan bir bayrak, hareket eden bir yürüyen merdiven ve yürüyen bir topluluk gibi tipik video süreçleridir. Yapay sahnelerin oluşturulması, özellikle beyaz perde endüstrisinde sahnelerinin beyaz bir perdeden parıldayan bir gerçekliğe zenginleştirilmesi için, çok önemli bir noktadır. Robot Bilim dünyasında, örneğin özerk hareket eden bir araç, üzerinden geçebileceği bir arazi (çim v.b.) ile geçemeyeceği bir araziyi (su v.b.) ayırt edebilmelidir. Bu problem örüntünün birkaç parçadan oluşan kategorilere, mesela çim, toprak, çalılar ve su gibi, sınıflandırılmasını adreslemektedir. Eğer bu parçalar teşhis edilebilirse, o zaman bu dokuların bölümlenmesi ve tanınması ile özerk hareket eden araç için etkili bir yol planlaması yapılabilir. Bu tezde, yukarıda bahsedilen bu dokulu hareketi nitelendirmeyi amaçladık. Bunun için bilinen çeşitli teknikleri uygulayarak birbirleri ile kıyasladık.

Anahtar Kelimeler: Hareketli Doku, Dinamik Doku, Doku Tanıma, Doku Sınıflama

TO MY MOTHER

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to Assoc. Prof. Dr. Gözde Bozdağı Akar for her guidance, valuable suggestions and insight throughout this research. I feeled great honour to know and work with her. It was a great pleasure for me everytime to see her in smiling face and optimistic.

Special thanks to Dr. Ünal Koyaz for his sincere helps.

I would also thank to Mustafa Akbostancı for his high tolerance and to İsmet Atalar for his understandings.

Many thanks to my friend Mehmet Karaduman for sharing his laptop for my study and I'am grateful to him for closing my absences in work.

I would also pleased to thank my company ASELSAN Inc. for supplying me an upright and comfortable working environment.

I apologize from my life for being absent in many occasions; sorry because I was just not proper at those times ...

I would like to extend my special appreciation and gratitude to MOM.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AR	Auto-Regressive
ARMA	Auto-Regressive Moving Average
IID	Independent and Identically Distributed
MA	Moving Average
MIT	Massachusetts Institute of Technology
PCA	Principal Component Analysis
SVD	Singular Value Decomposition

CHAPTER 1

INTRODUCTION

In robotics, robot vision is the most popular and growing research area with many of problems remaining unsolved. Completely automated designs are the ones that we aimed to reach for realizing specialized purposes. For example, one can design a machine for garden arrangement. The machine will cut out the grasses, give water to the flowers or trim the bushes. Another complicated example, one can need a fully autonomous vehicle which only needs world coordinates to travel around to collect information for some purposes. A machine like this has to handle a huge task compared with the garden arrangement machine. However the tasks between these machines are so different, each of them must know the right and the wrong. The garden arrangement machine has to distinguish flowers from grasses for not cutting them out instead of giving them water. And the autonomous vehicle has to distinguish road from river for travelling on the proper surface. Once we have a chance to divide and characterize the world into identifiable portions of images then working with machines like these are not so far beyond now.

The problem is to segment a frame into meaningful portions of images, but it must be learned before what are to be segmented which is the subject that we are interested. It is obvious that the data will be consist of continuous frames of images. So, our segmentation problem is not stationary. The focus of segmentation lies under defining the motions within the frames. Of course the world is full of motion that can not be defined with tens or hundreds or thousands of known motions. Here the motions subject for our thesis are the textured motions. In literature the subject is mentioned as textured motion [32], temporal texture [37] and dynamic textures [1]. We prefer using dynamic textures in this thesis.

What is a dynamic texture? How can we distinguish dynamic textures? A person walking or swimming, or the spinning wheels of a machine, are defined as examples of activities in [8, 29, 42, 43], because these kind of motions are temporally periodic but spatially restricted. What else for motions such as opening a door, lifting a briefcase, or throwing a ball? This class of motions are motion events which are single motions that do not repeat spatially or temporally [9, 37]. Then what is dynamic texture? Wavy water, rising smoke, falling snow, flock of birds flying, river waves, crowd of people walking, a moving escalator, burning fire, etc. are some samples for dynamic textures.

Dynamic textures rised research areas especially on synthesis and recognition. Synthesis of dynamic textures is so useful for real like generation of synthetic frames as it can be observed in many Hollywood productions. For example, a war scene can be produced with only a few hundred people which can be presented as a war of two country each having thousands of soldiers in the field. However the synthesis works are well studied that they are in use for real life needs, the recognition of dynamic textures is a new and highly challenging problem, and there exist a growing research on this area.

1.1 Motivation

From an applications point of view, the major motivation for working on dynamic textures is the advent of large video databases. There are a huge amount of video in digital form. It is important to search and handle these databases while modeling or recognizing the dynamic textures. In recent years, with the increasing processing power of computers, growing of memory and storage capacities, it is now simpler than ever to compute these tasks. This growing up of computational technology also made researchers to boom their interests on these subjects.

In this thesis, our main goal is to compare the well-known dynamic texture classification techniques that have confirmed their success in literature. It can be concluded that it is possible to collect these dynamic texture classification techniques in three groups : ones use spatial information, ones use temporal information and finally ones use both spatial and temporal information. When we had investigated the literature we saw that the ones using both spatial and temporal information have better recognition rates. These techniques which use both spatial and temporal information can also be collected into two groups; by the way that the solution is seeked: stochastic or deterministic.

In this thesis, a comparison of stochastic and deterministic techniques is aimed. Before comparison, we have given a detailed analysis of the subject. During our analysis, we have dealed with defining regularities of textures with deterministic methods, synthesis of dynamic textures with stochastic methods and finally classification of dynamic textures with both stochastic and deterministic methods.

We have aimed to compare two methods one using stochastic techniques referring to [1, 2, 3] and one using deterministic techniques referring to [4, 5, 6, 7, 23].

1.2 Outline of the Thesis

In chapter 2, a brief literature survey on dynamic textures is given.

In chapter 3, some of the basics that we have used in our study can be found.

In chapter 4, a detailed explanation of stochastic method is presented.

In chapter 5, a detailed explanation of deterministic method is presented.

In chapter 6, implementation is explained, clues of calculations are given and results are shown.

And finally, chapter 7 concludes this thesis.

CHAPTER 2

LITERATURE SURVEY

In this chapter, a literature survey is presented on textures and dynamic textures.

2.1 Texture

In computer vision, texture definitions appeared relatively a long time ago, nearly when the new era on computations give rise with the evolution of computers. In literature, we can find some good definitions for texture. Here are some definitions for texture :

"The term texture generally refers to repetition of basic texture elements called texels. The texel contains several pixels, whose placement could be periodic, quasi-periodic or random. Natural textures are generally random, whereas artificial textures are often deterministic or periodic. Texture may be coarse, fine, smooth, granulated, rippled, regular, irregular, or linear." [62].

"Textured regions are spatially extended patterns based on the more or less accurate repetition of some unit cell (texton or subpattern)." [63].

"Textures are homogeneous patterns or spatial arrangements of pixels that regional intensity or color alone does not sufficiently describe. As such, textures have statistical properties, structural properties, or both. They may consist of the structured and/or random placement of elements, but also may be without fundamental subunits." [64].

"A Texture is a signal that exhibit the following property. Using any window of size larger than some critical size, the "information content" exhibited in the window is invariant to the window's position within the given sample." [35].

As we can easily observe, life is full of textures. Roads, trees, clouds, water, rivers, etc. are all textures that we can come accross everyday. A cell of any living organism, blood, bones, iris of eye, etc. are also textures. By growing usage of computers in our life in recent decades, many research groups interested with textures. Researchers dealed with detecting, comparing, tracking, segmenting, synthesizing, classifying problems of texture for different purposes. While some of them tried to catch change of texture and investigate the results for solving a disease, some of them tried to expand a texture from a sample of it for creating synthetic textures. Interested readers on synthesis of textures are refered to [44, 45, 46, 47, 48, 49, 50, 51, 52].

2.2 Dynamic Texture

In simple words, dynamic texture is a video which consists of texture images. A flock of bird flying, a river flowing, a fire burning, a water boiling, a tree waving with the blowing wind, etc. are all examples of dynamic textures.

While textures are being investigated by research groups for a long time, studies on dynamic textures are relatively novel. With increasing computation power in recent years, researchers boom their interest on dynamic textures. Researchs on dynamic textures intensify especially on segmentation, synthesis and recognition subjects.

2.3 Previous Work

Dynamic texture synthesis is being studied especially in the field of computer graphics. Synthesis of texture movie arouse interest of industry working on special effects for motion pictures and television, computer-generated animation, computer games and computer art. Synthesis allows creation of synthetic textures in any size, long and behavior. Some good examples on dynamic texture synthesis can be investigated in papers [1] and [32].

As we focused on dynamic texture recognition, we see that the first occurence of subject in literature dates back to early nineties with the published paper [8]. In these first studies as explained in [13], visual motion is categorised into three classes : activities, motion events and temporal textures. Activities, such as walking or digging, are defined as motion patterns that are periodic in time and localized in space. Motion events, like opening a door, do not show temporal or spatial periodicity. Finally, temporal textures exhibit statistical regularity but have indeterminate spatial and temporal extent. When we investigate studies on recent years, as stated in [13], the existing approaches to temporal texture recognition can be classified into one of the following groups: methods based on optic flow [4, 7, 24, 29, 65, 66, 67, 68, 69, 70], methods computing geometric properties in the spatiotemporal domain [25, 71], methods based on local spatiotemporal filtering [72], methods using global spatiotemporal transforms [73] and, finally, model-based methods that use estimated model parameters as features [19, 30, 45, 74, 75, 76]. Methods based on optic flow are currently the most popular because optic flow estimation is a computationally efficient and natural way to characterize the local dynamics of a temporal texture. It helps reduce dynamic texture analysis to analysis of a sequence of instantaneous motion patterns viewed as static textures. When necessary, image texture features can be added to the motion features, to form a complete feature set for motion and appearance-based recognition.

A good example for model-based methods is described in [1], and other good example for dynamic texture recognition depending on optical flow and texture features is described in [4], which are also pioneered the studies explained in this thesis. Details on these studies are given in chapter 4 and chapter 5 of this thesis.

CHAPTER 3

BASICS

In this chapter, brief information on some of the basics concerning to our work are given.

3.1 Principal Components Analysis

Information in this section is given referring to [58].

In statistics, principal components analysis (PCA) is a technique that can be used to simplify a dataset; more formally it is a linear transformation that chooses a new coordinate system for the data set such that the greatest variance by any projection of the data set comes to lie on the first axis (then called the first principal component), the second greatest variance on the second axis, and so on. PCA can be used for reducing dimensionality in a dataset while retaining those characteristics of the dataset that contribute most to its variance by eliminating the later principal components (by a more or less heuristic decision). These characteristics may be the "most important", but this is not necessarily the case, depending on the application.

PCA is also called the Karhunen-Loève transform or the Hotelling transform. PCA has the speciality of being the optimal linear transformation for keeping the subspace that has largest variance. However this comes at the price of greater computational requirement, e.g. if compared to the discrete cosine transform. Unlike other linear transforms, the PCA does not have a fixed set of basis vectors. Its basis vectors depend on the data set.

Assuming zero empirical mean (the empirical mean of the distribution has been subtracted away from the data set), the principal component w_1 of a dataset x can be defined as:

$$w_1 = \arg\max_{\|w\|=1} E\left\{\left(w^T x\right)^2\right\} \tag{3.1}$$

With the first $k-1$ components, the k^{th} component can be found by subtracting the first $k-1$ principal components from x:

$$\hat{x}_{k-1} = x - \sum_{i=1}^{k-1} w_i w_i^T x \tag{3.2}$$

and by substituting this as the new dataset to find a principal component in

$$w_k = \arg\max_{\|w\|=1} E\left\{\left(w^T \hat{x}_{k-1}\right)^2\right\} \tag{3.3}$$

A simpler way to calculate the components w_i uses the empirical covariance matrix of x, the measurement vector. By finding the eigenvalues and eigenvectors of the covariance matrix, we find that the eigenvectors with the largest eigenvalues correspond to the dimensions that have the strongest correlation in the dataset. The original measurements are finally projected onto the reduced vector space. Note that the eigenvectors X are actually the columns of the matrix V, where $X = ULV'$ is the singular value decomposition of X.

PCA is a popular technique in pattern recognition. PCA optimally minimizes reconstruction error under the L^2 norm.

3.1.1 Algorithm Details

Following is a detailed description of PCA using the covariance method. Suppose you have n data vectors $x_1 \ldots x_n$ each length d, written as $x_m = (x_m^1 \ldots x_m^d)$, and you want to project your data into a k dimensional subspace.

3.1.1.1 Find the basis vectors

1. Organize your data into column vectors, so you end up with a dxn matrix, D.
2. Find the empirical mean along each dimension, so you end up with a $dx1$ empirical mean vector, M.

3. Subtract the empirical mean vector M from each column of the data matrix D. Store mean-subtracted data dxn matrix in S.
4. Find the empirical covariance dxd matrix C of S. $C = SS^T$.
5. Compute and sort by decreasing eigenvalue, the eigenvectors V of C.
6. Save the mean vector M. Save the first k columns of V as P. P will have dimension dxk, $1 \le k \le d$.

3.1.1.2 Observation

After computing the matrix C with elements $C_{i,j} = \sum_{m=1}^{n} x_m^i x_m^j$, we can extract from the diagonal $C_{ii} = (n-1)(\sigma^i)^2$, and compute the correlation matrix R with $R_{ij} = C_{ij} / (\sigma^i \sigma^j)$. The matrix R is symmetric (like the covariance matrix), its values are between -1 and 1, and the diagonal contains n times the value 1. When the different dimensions of the input data have different measuring units, by using the matrix C we are computing linear combinations of data of different scales; thus using the normalized matrix R makes more sense.

In that case steps 5 and 6 become:

5. Compute and sort by decreasing eigenvalue, the eigenvectors V of R.
6. Save the mean vector M and $\sigma = (\sigma^1 \ldots \sigma^d)$. Save the first k columns of V as P. P will have dimension dxk, $1 \le k \le d$.

3.1.2 Projecting New Data

Suppose you have a $dx1$ data vector D. Then the $kx1$ projected vector is $v = P^T(D-M)$.

If the correlation matrix R has been used instead of the covariance matrix C, the elements of the input vector should be normalized: $Z^i = (D^i - M^i)(\sigma^i)^{-1}$. Then the projected vector is $v = P^T Z$.

3.1.3 Derivation of PCA using the Covariance Method

Let X be a d dimensional random vector expressed as column vector. Without loss of generality, assume X has zero empirical mean. We want to find a dxd orthonormal projection matrix P such that $Y = P^T X$ with the constraint that $\operatorname{cov}(Y)$ is a diagonal matrix and $P^{-1} = P^T$.

By substitution, and matrix algebra, we obtain:

$$\begin{aligned} \operatorname{cov}(Y) &= E\left[YY^T\right] \\ &= E\left[(P^T X)(P^T X)^T\right] \\ &= E\left[(P^T X)(X^T P)\right] \\ &= P^T E\left[XX^T\right]P \\ &= P^T \operatorname{cov}(X)P \end{aligned}$$. We now have: $$\begin{aligned} P\operatorname{cov}(Y) &= PP^T \operatorname{cov}(X)P \\ &= \operatorname{cov}(X)P \end{aligned}$$.

Rewrite P as $dx1$ column vectors, so $P = \left[P_1, P_2, \cdots, P_d\right]$ and $\operatorname{cov}(Y)$ as:

$$\begin{bmatrix} \lambda_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_d \end{bmatrix}$$. Substituting into equation above, we obtain:

$\left[\lambda_1 P_1, \lambda_2 P_2, \ldots, \lambda_d P_d\right] = \left[\operatorname{cov}(X)P_1, \operatorname{cov}(X)P_2, \ldots, \operatorname{cov}(X)P_d\right]$. Notice that in $\lambda_i P_i = \operatorname{cov}(X)P_i$, P_i is an eigenvector of X's covariance matrix. Therefore, by finding the eigenvectors of X's covariance matrix, we find a projection matrix P that satisfies the original constraints.

3.2 Singular Value Decomposition

Information in this section is given referring to [58].

In linear algebra singular value decomposition (SVD) is an important factorization of a rectangular real or complex matrix, with several applications in signal processing and statistics. This matrix decomposition is analogous to the diagonalization of symmetric or Hermitian square matrices using a basis of eigenvectors given by the spectral theorem.

3.2.1 Statement of the Theorem

Suppose M is an mxn matrix whose entries come from the field K, which is either the field of real numbers or the field of complex numbers. Then there exists a factorization of the form

$$M = U\Sigma V^* \tag{3.4}$$

where U is an mxm unitary matrix over K, the matrix Σ is mxn with non-negative numbers on the diagonal and zeros off the diagonal, and V^* denotes the conjugate transpose of V, an nxn unitary matrix over K. Such a factorization is called a singular-value decomposition of M.

- The matrix U describes the rows of M with respect to the base vectors associated with the singular values.
- The matrix V describes the columns of M with respect to the base vectors associated with the singular values.
- The matrix Σ contains the singular values.

One commonly insists that the values $\Sigma_{i,j}$ be ordered in non-increasing fashion. In this case, the diagonal matrix Σ is uniquely determined by M (though the matrices U and V are not).

3.2.2 Singular Values, Singular Vectors, Relation to the SVD

A non-negative real number σ is a singular value for M if there exist non-zero vectors u in K^m and v in K^n such that $Mv = \sigma u$ and $M^*u = \sigma v$. The vectors u and v are called right-singular and left-singular vectors for σ, respectively.

In any singular value decomposition $M = U\Sigma V^*$ the diagonal entries of Σ are necessarily equal to the singular values of M. The columns of U and V are left and right, respectively, singular vectors for the corresponding singular values. Note that the singular vectors are not uniquely determined by a given singular value, and likewise the matrices U and V are not uniquely determined by the matrix M.

One can show that the non-zero singular values for M are precisely the square roots of the non-zero eigenvalues of the positive semi-definite matrix MM^*, and these are precisely the square roots of the non-zero eigenvalues of M^*M. Furthermore, the columns of U are eigenvectors of MM^* and the columns of V are eigenvectors of M^*M.

3.2.3 Geometric Meaning

Because U and V are unitary, we know that the columns $u_1,\ldots,u_m$ of U yield an orthonormal basis of K^m and the columns $v_1,\ldots,v_n$ of V yield an orthonormal basis of K^n (with respect to the standard scalar products on these spaces).

The linear transformation $T: K^n \rightarrow K^m$ that takes a vector x to Mx has a particularly simple description with respect to these orthonormal bases: we have $T(v_i) = \sigma_i u_i$, for $i = 1,\ldots,\min(m,n)$, where σ_i is the i^{th} diagonal entry of Σ, and $T(v_i) = 0$ for $i > \min(m,n)$.

The geometric content of the SVD theorem can thus be summarized as follows: for every linear map $T: K^n \rightarrow K^m$ one can find orthonormal bases of K^n and K^m such that T maps the i^{th} basis vector of K^n to a non-negative multiple of the i^{th} basis vector of K^m, and sends the left-over basis vectors to zero. With respect to these bases, the map T is therefore represented by a diagonal matrix with non-negative real diagonal entries.

3.2.4 Applications of the SVD

The singular value decomposition is used for computing the pseudoinverse of a matrix. Indeed, the pseudoinverse of the matrix M with singular value decomposition $M = U\Sigma V^*$ is

$$M^{+} = V\Sigma^{+}U^{*} \tag{3.5}$$

where Σ^{+} is the transpose of Σ with every non-zero entry replaced by its reciprocal. The pseudoinverse is needed to solve linear least squares problems.

The SVD is also applied extensively to the study of linear inverse problems, and is useful in the analysis or regularization methods such as that of Tikhonov. It is widely used in statistics where it is related to principal component analysis, and in signal processing and pattern recognition. It is also used in output-only modal analysis, where the non-scaled mode shapes can be determined from the singular vectors.

3.3 Eigenvalues

Information in this section is given referring to [58].

In mathematics, an eigenvector of a transformation is a vector whose direction is unchanged by that transformation. The factor by which the magnitude is scaled is called the eigenvalue of that vector. A pictorial example is provided in Figure-3.1. Often, a transformation is completely described by its eigenvalues and eigenvectors. An eigenspace is a set of eigenvectors with the same eigenvalue.

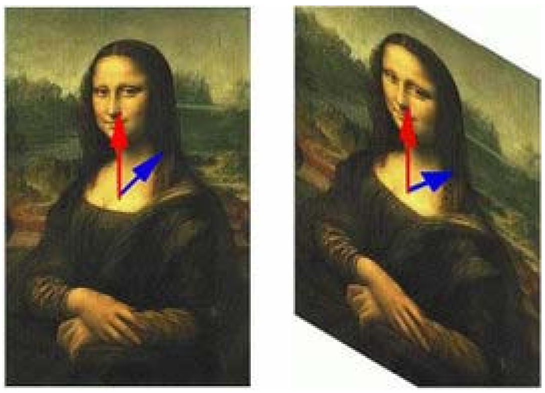


Figure 3-1: *A pictorial example for eigenvalues, eigenvectors and eigenspace.*

In this shear transformation of the Mona Lisa, shown in Figure-3.1, the picture was deformed in such a way that its central vertical axis was not modified. (Note: The corners have been cropped on the right hand picture.) The blue vector, from her chest to her shoulder, has changed direction, but the red one, from her chest to her chin, is unchanged. The red vector is thus an eigenvector of the transformation and the blue vector is not. Since the red vector was neither stretched nor compressed, its eigenvalue is 1. All vectors along the same vertical line are also eigenvectors, with the same eigenvalue. They form the eigenspace for this eigenvalue.

For an example; as the Earth rotates, every arrow pointing outward from the center of the Earth also rotates, except those arrows that lie on the axis of rotation. Consider the transformation of the Earth after one hour of rotation: An arrow from the center of the Earth to the Geographic South Pole would be an eigenvector of this transformation, but an arrow from the center of the Earth to anywhere on the equator would not be an eigenvector. Since the arrow pointing at the pole is not stretched by the rotation of the Earth, its eigenvalue is 1.

3.3.1 Computing Eigenvalues of Matrices

Suppose that we want to compute the eigenvalues of a given matrix. If the matrix is small, we can compute them symbolically using the characteristic polynomial. However, this is often impossible for larger matrices, in which case we must use a numerical method.

An important tool for describing eigenvalues of square matrices is the characteristic polynomial: saying that λ is an eigenvalue of A is equivalent to stating that the system of linear equations $(A-\lambda I)v=0$ (where I is the identity matrix) has a non-zero solution v (an eigenvector), and so it is equivalent to the determinant: $\det(A-\lambda I)=0$.

The function $p(\lambda) = \det(A - \lambda I)$ is a polynomial in λ since determinants are defined as sums of products. This is the characteristic polynomial of A: the eigenvalues of a matrix are the zeros of its characteristic polynomial.

All the eigenvalues of a matrix A can be computed by solving the equation $p_A(\lambda) = 0$. If A is an nxn matrix, then p_A has degree n and A can therefore have at most n eigenvalues. Conversely, the fundamental theorem of algebra says that this equation has exactly n roots (zeros), counted with multiplicity. All real polynomials of odd degree have a real number as a root, so for odd n, every real matrix has at least one real eigenvalue. In the case of a real matrix, for even and odd n, the non-real eigenvalues come in conjugate pairs.

Once the eigenvalues λ are known, the eigenvectors can then be found by solving: $(A - \lambda I)v = 0$.

3.4 Optical Flow

Information in this section is given referring to [61].

Motion estimation is an important part of any video processing system. All the motion estimation algorithms are based on temporal changes in image intensities (more generally color). In fact, the observed 2D motions based on intensity changes may not be the same as the actual 2D motions. To be more precise, the velocity of observed or apparent 2D motion vectors are referred to as optical flow. Optical flow can be caused not only by object motions, but also camera movements or illumination condition changes.

3.4.1 2D Motion vs. Optical Flow

The human eye perceives motion by identifying corresponding points at different times. The correspondence is usually determined by assuming that the color or brightness of a point does not change after the motion. It is interesting to note that the observed 2D motion can be different from the actual projected 2D motion under certain circumstances. Figure-3.2

illustrates two special cases. In the first example, a sphere with a uniform at surface is rotating under a constant ambient light. Because every point on the sphere reflects the same color, the eye cannot observe any change in the color pattern of the imaged sphere and thus considers the sphere as being stationary. In the second example, the sphere is stationary, but is illuminated by a point light source that is rotating around the sphere. The motion of the light source causes the movement of the reflecting light spot on the sphere, which in turn can make the eye believe the sphere is rotating. The observed or apparent 2D motion is referred to as optical flow in computer vision literature.

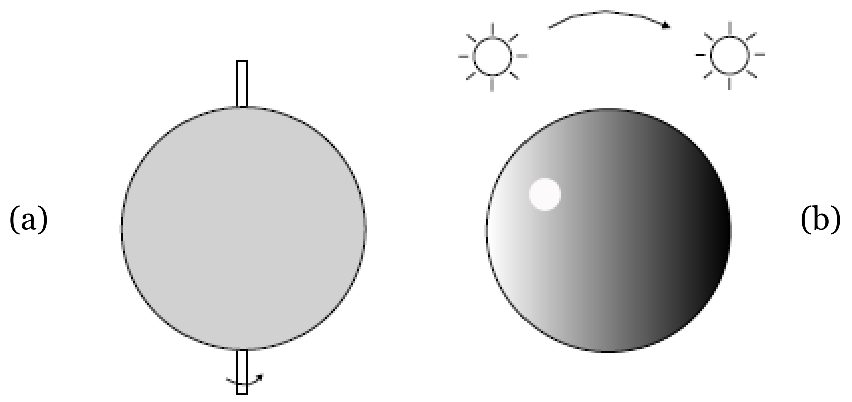


Figure 3-2: *The optical flow is not always the same as the true motion field. In (a), a sphere is rotating under a constant ambient illumination, but the observed image does not change. In (b), a point light source is rotating around a stationary sphere, causing the highlight point on the sphere to rotate.*

The above examples reveal that the optical flow may not be the same as the true 2D motion. When only image color information is available, the best one can hope to estimate accurately is the optical flow. However, in the remaining part of this section, we will use the term 2D motion or simply motion to describe optical flow. The readers should bear in mind that sometimes it may be different from the true 2D motion.

3.4.2 Optical Flow Equation and Ambiguity in Motion Estimation

Consider a video sequence whose luminance variation is represented by $\psi(x,y,t)$. Suppose an imaged point (x,y) at time t is moved to $(x+d_x, y+d_y)$ at time $t+d_t$. Under the constant intensity assumption, the images of the same object point at different times have the same luminance value. Therefore,

$$\psi(x+d_x, y+d_y, t+d_t) = \psi(x,y,t) \tag{3.6}$$

Using Taylor's expansion, when d_x, d_y, d_t are small, we have

$$\psi(x+d_x, y+d_y, t+d_t) = \psi(x,y,t) + \frac{\partial\psi}{\partial x}d_x + \frac{\partial\psi}{\partial y}d_y + \frac{\partial\psi}{\partial t}d_t \tag{3.7}$$

Combining equations (3.6) and (3.7) yields

$$\frac{\partial\psi}{\partial x}d_x + \frac{\partial\psi}{\partial y}d_y + \frac{\partial\psi}{\partial t}d_t = 0 \tag{3.8}$$

The above equation is written in terms of the motion vector (d_x, d_y). Dividing both sides by d_t yields

$$\frac{\partial\psi}{\partial x}\upsilon_x + \frac{\partial\psi}{\partial y}\upsilon_y + \frac{\partial\psi}{\partial t} = 0 \text{ or } \nabla\psi^T v + \frac{\partial\psi}{\partial t} = 0 \tag{3.9}$$

where (υ_x, υ_y) represents the velocity vector, $\nabla\psi = \left[\frac{\partial\psi}{\partial x}, \frac{\partial\psi}{\partial y}\right]^T$ is the spatial gradient vector of $\psi(x,y,t)$. In arriving at the above equation, we have assumed that d_t is small, so that $\upsilon_x = d_x / d_t$, $\upsilon_y = d_y / d_t$. The above equation is commonly known as the optical flow equation.

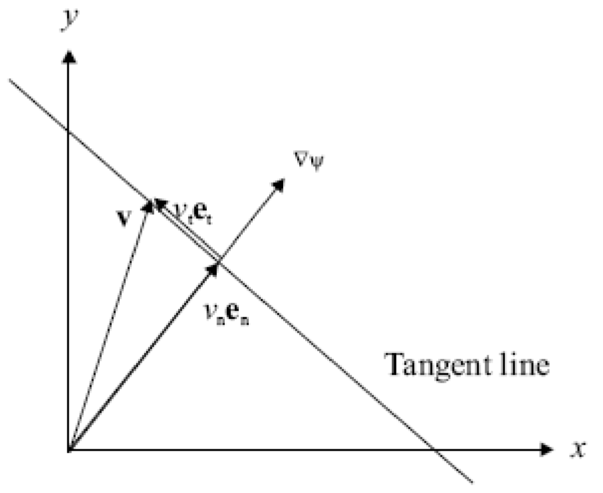


Figure 3-3: *Decomposition of motion v into normal $(\upsilon_n e_n)$ and tangent $(\upsilon_t e_t)$ components. Given $\nabla\psi$ and $\frac{\partial\psi}{\partial t}$, any motion vector on the tangent line satisfies the optical flow equation.*

As shown in Figure-3.3, the flow vector v at any point x can be decomposed into two orthogonal components as

$$v = \upsilon_n e_n + \upsilon_t e_t \tag{3.10}$$

where e_n is the direction vector of the image gradient $\nabla\psi$, to be called the normal direction, and e_t is orthogonal to e_n, to be called the tangent direction. The optical flow equation in equation (3.9) can be written as

$$\upsilon_n \|\nabla\psi\| + \frac{\partial\psi}{\partial t} = 0 \tag{3.11}$$

where $\|\nabla\psi\|$ is the magnitude of the gradient vector. Three consequences from equation (3.9) or (3.11) are :

1. At any pixel x, one cannot determine the motion vector v based on $\nabla\psi$ and $\frac{\partial\psi}{\partial t}$ alone. There is only one equation for two unknowns (υ_x and υ_y, or υ_n and υ_t). In fact, the underdetermined component

is υ_t. To solve both unknowns, one needs to impose additional constraints. The most common constraint is that the flow vectors should vary smoothly spatially, so that one can make use of the intensity variation over a small neighborhood surrounding x to estimate the motion at x.

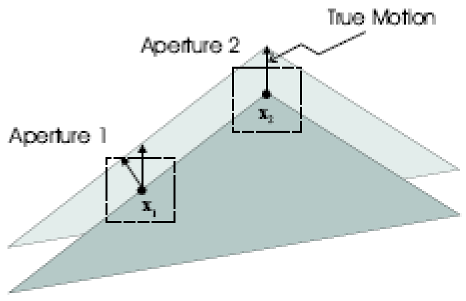


Figure 3-4: *The aperture problem in motion estimation : To estimate the motion at x_1 using aperture 1, it is impossible to determine whether the motion is upward or perpendicular to the edge, because there is only one spatial gradient direction in this aperture. On the other hand, the motion at x_2 can be determined accurately, because the image has gradient in two different directions in aperture 2.*

2. Given $\nabla\psi$ and $\frac{\partial\psi}{\partial t}$, the projection of the motion vector along the normal direction is fixed, with $\upsilon_n = -\frac{\partial\psi}{\partial t}/\|\nabla\psi\|$, whereas the projection onto the tangent direction, υ_t, is undetermined. Any value of υ_t would satisfy the optical flow equation. In Figure-3.3, this means that any point on the tangent line will satisfy the optical flow equation. This ambiguity in estimating the motion vector is known as the aperture problem. The word “aperture” here refers to the small window over which to apply the constant intensity assumption. The motion can be estimated uniquely only if the

aperture contains at least two different gradient directions, as illustrated in Figure-3.4.

3. In regions with constant brightness so that $\|\nabla \psi\| = 0$, the flow vector is indeterminate. This is because there is no perceived brightness changes when the underlying surface has a flat pattern. The estimation of motion is reliable only in regions with brightness variation, i.e., regions with edges or non-flat textures.

3.5 Horn-Schunk Method

Horn-Schunk [6] method is one of the best computation techniques for estimating motion vectors. Details of this method will be introduced in this section.

Let $I(x,y,t)$ be image brightness at point (x,y) at time t and $I_x = \frac{\partial I}{\partial x}$, $I_y = \frac{\partial I}{\partial y}$, $I_t = \frac{\partial I}{\partial t}$, $u = \frac{d_x}{d_t}$, $v = \frac{d_y}{d_t}$. As described in section 3.4, we know that $uI_x + vI_y + I_t = 0$. With these definitions step by step explanation of the method :

1. Define an energy function and minimize,

$$E(x,y) = (uI_x + vI_y + I_t)^2 + \lambda f \text{, where } f = u_x^2 + u_y^2 + v_x^2 + v_y^2 \qquad (3.12)$$

2. Differentiate with respect to unknowns u and v,

$$\frac{\partial E}{\partial u} = 2I_x(uI_x + vI_y + I_t) + \frac{\partial f}{\partial u} \text{, where } \frac{\partial f}{\partial u} = \frac{\partial}{\partial u}\frac{\partial u}{\partial x} + \frac{\partial}{\partial u}\frac{\partial u}{\partial y} = 2(u_{xx} + u_{yy}) \qquad (3.13)$$

$$\frac{\partial E}{\partial v} = 2I_y(uI_x + vI_y + I_t) + \frac{\partial f}{\partial v} \text{, where } \frac{\partial f}{\partial v} = \frac{\partial}{\partial v}\frac{\partial v}{\partial x} + \frac{\partial}{\partial v}\frac{\partial v}{\partial y} = 2(v_{xx} + v_{yy}) \qquad (3.14)$$

In equations above, $(u_{xx} + u_{yy})$ and $(v_{xx} + v_{yy})$ are the laplacians of u and v, respectively. Thus, we have

$$I_x(uI_x + vI_y + I_t) + \Delta^2 u = 0 \tag{3.15}$$

$$I_y(uI_x + vI_y + I_t) + \Delta^2 v = 0 \tag{3.16}$$

Laplacian controls smoothness of optical flow. A particular choice can be $\Delta^2 u = u - u_{avg}$, $\Delta^2 v = v - v_{avg}$. With rearranging equations, we have :

$$u(\lambda + I_x^2) + vI_xI_y + I_xI_t - \lambda u_{avg} = 0 \tag{3.17}$$

$$v(\lambda + I_y^2) + uI_xI_y + I_yI_t - \lambda v_{avg} = 0 \tag{3.18}$$

We have two equations ((3.17) and (3.18)) and two unknowns. Writing v in terms of u and plugging it in other equation, we obtain :

$$u = u_{avg} - I_x\left(\frac{I_xu_{avg} + I_yv_{avg} + I_t}{I_x^2 + I_y^2 + \lambda}\right) \tag{3.19}$$

$$v = v_{avg} - I_y\left(\frac{I_xu_{avg} + I_yv_{avg} + I_t}{I_x^2 + I_y^2 + \lambda}\right) \tag{3.20}$$

3. Now, iteratively compute u and v, assuming that initially u and v are 0. Compute u_{avg} and v_{avg} in a neighborhood.

3.5.1 Estimation Hints

In this section, hints on estimating I_x, I_y, I_t and u, v are given which are needed while computing optical flow.

$$\begin{aligned}I_x(i,j,k) = (&I(i,j+1,k) - I(i,j,k) + I(i+1,j+1,k) - I(i+1,j,k) + \\ &I(i,j+1,k+1) - I(i,j,k+1) + I(i+1,j+1,k+1) - I(i+1,j,k+1))/4,\end{aligned} \tag{3.21}$$

$$\begin{aligned}I_y(i,j,k) = (&I(i+1,j,k) - I(i,j,k) + I(i+1,j+1,k) - I(i,j+1,k) + \\ &I(i+1,j,k+1) - I(i,j,k+1) + I(i+1,j+1,k+1) - I(i,j+1,k+1))/4,\end{aligned} \tag{3.22}$$

$$\begin{aligned}I_t(i,j,k) = (&I(i,j,k+1) - I(i,j,k) + I(i+1,j,k+1) - I(i+1,j,k) + \\ &I(i,j+1,k+1) - I(i,j+1,k) + I(i+1,j+1,k+1) - I(i+1,j+1,k))/4.\end{aligned} \tag{3.23}$$

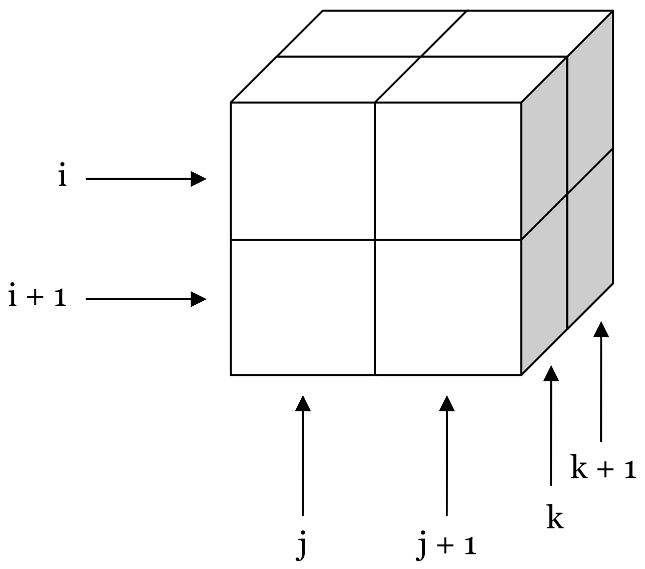


Figure 3-5: *Index references for pixels of consecutive frames of a video defined on a cubic presentation, i for rows, j for columns and k for frames.*

$$\bar{u}(i,j) = (u(i-1,j) + u(i+1,j) + u(i,j-1) + u(i,j+1))/4 \tag{3.24}$$

$$\bar{v}(i,j) = (v(i-1,j) + v(i+1,j) + v(i,j-1) + v(i,j+1))/4 \tag{3.25}$$

While computing, special cases at edges and corners must be taken into account for equations (3.21) to (3.25).

CHAPTER 4

STOCHASTIC APPROACH

In this chapter, the stochastic solution to dynamic texture classification problem is defined. The solution depends on the paper by Doretto [1]. This paper is selected for pioneering our work because of its reported success on dynamic texture classification. Other reasons for our selection are proposed method's ability on synthesis, compression and its flexibility on editing dynamic textures. However all of these attributes are realizable thanks to modeling, we will be concantrate on proposed method's approach to classification. Interested readers can also refer to papers [2, 3, 11, 12, 17, 18, 19, 20, 30, 31, 36] and books [54, 55, 56] for constituting the best knowledge about the subject.

4.1 Modeling

In [77], Zhu describes texture as a realization from a stationary stochastic process with spatially invariant statistics. This definition captures the intuitive notion of texture. For a sequence of images (time-varying texture), individual images are clearly not independent realizations from a stationary distribution, for there is a temporal coherence intrinsic in the process that needs to be captured. Therefore, individual images can be modeled as realizations of the output of a dynamical system driven by an independent and identically distributed (IID) process.

4.1.1 A State-Space Model

In modeling of a dynamic texture, we will start with the capturing process. In a real application it is obvious that the sequence of images must be captured by an imaging device. While this capturization it is clearly known that the final image is a noisy version of the original image where the noise is inferred from the physics of the imaging device. In Figure-4.1 this phenomenon is represented. Let $y(t) = i(t) + w(t)$ is the measured frame at time t, where $i(t)$ is the original image and $w(t)$ is the noise which is an IID drawn from a known distribution. The measured sequence will be $Y = I + W$, in matrix notation, which will be named as the space model of the dynamic texture. In [1], this phenomenon is aimed to be modeled so that the dynamic textures can be represented in words of parameters. The target modeling is aimed to be able to capture both spatial and temporal aspects of the textures. The model is aimed to be applicable for recognition tasks, such as identifying a given video sequence as belonging to the wavy or to a motionless water class. It is also aimed to be suitable for segmentation tasks, in other words, for partitioning a video sequence into homogeneous regions.

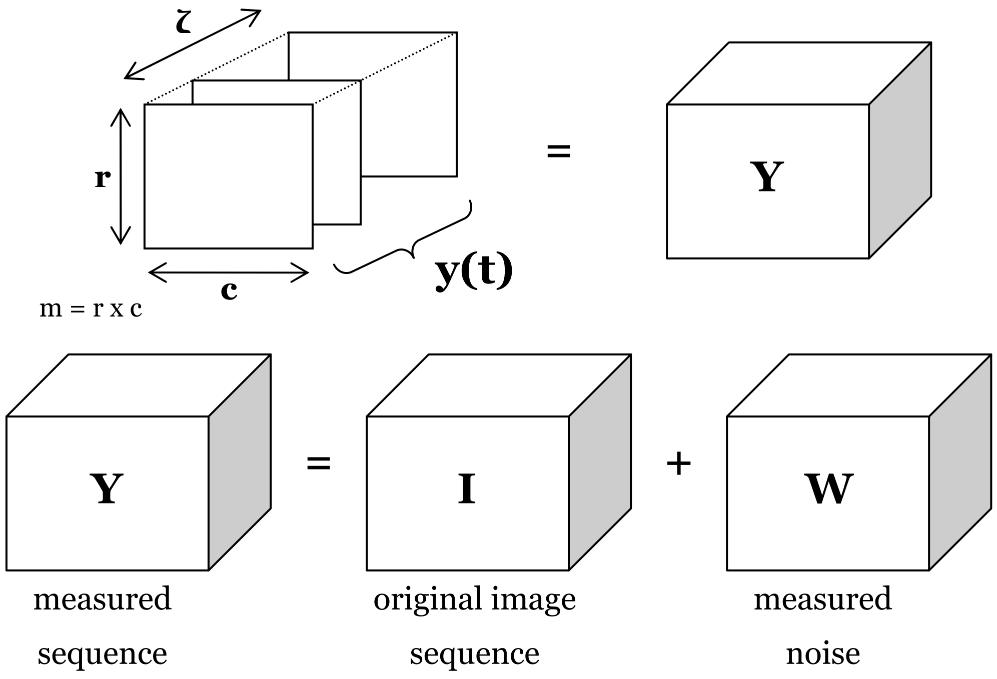


Figure 4-1: *Space model of a dynamic texture*

For making the definitions and equations more comprehensible, it is declared here that the capital letters Y, I, W will be used for representing sequences in matrix notation, while small letters y, i, w representing single frames.

Following the capturing process we have an image sequence, $Y = I + W$. In Figure-4.1, it is emphasized that the measured sequence Y is the noisy version of original image sequence, dynamic texture, I which is aimed to be modeled. Dynamic textures are modeled as auto-regressive moving average processes (ARMA) as described in [1]. The model forms (linear) dynamic textures with combination of spatial filters and state components. Let, there exists a set of n spatial filters $\phi_\alpha : \Re \rightarrow \Re^m, \alpha = 1 \ldots n$, $i(t) = \phi(x(t))$, where $\phi(\cdot)$ indicates the combination of the output of the n filters $\{\phi_\alpha\}$ respectively applied to each of the n state components, $x(t)$. As the model is aimed to associate a dynamic texture to an ARMA process, AR (auto-regressive) and MA (moving average) coefficients are chosen with order 1, without loss of generality. So, we have $x(t+1) = Ax(t) + Bv(t)$ with $v(t) \in \Re^{n_v}$ an unknown IID realization, initial condition $x(0) = x_0$ and $A \in \Re^{nxn}, B \in \Re^{nxn_v}$, AR and MA coefficients defining the ARMA process, respectively. Therefore, a linear dynamic texture is associated to an ARMA process with unknown input distribution

$$\begin{cases} x(t+1) = Ax(t) + Bv(t) \\ y(t) = \phi(x(t)) + w(t) \end{cases} \tag{4.1}$$

with $x(0) = x_0$, $v(t)$: an IID realization unknown, $w(t)$: an IID realization given, such that $i(t) = \phi(x(t))$.

As stated in [1] equation (4.1) is the ARMA model for representing dynamic textures. Following the definition of dynamic textures as an ARMA model, the choice of filters is so important that the filters will be the part of the learning process for a given dynamic texture and will supply a dimensionality reduction. So that the filters are chosen as a decomposition of the image in the simple (linear) form

$$i(t) = \phi(x(t)) = \sum_{i=1}^{n} x_i(t)\theta_i \doteq Cx(t) \qquad (4.2)$$

where $C = [\theta_1, \ldots, \theta_n] \in \Re^{mxn}$ and $\{\theta_i\}$ can be an orthonormal basis of L^2, a set of principal components, or a wavelet filter bank. In our implementations, $\{\theta_i\}$ is selected as a set of principal components. In equation (4.2), notice that, C represents the spatial filters and $x(t)$ represents the state components. This decomposition is represented in Figure-4.2.

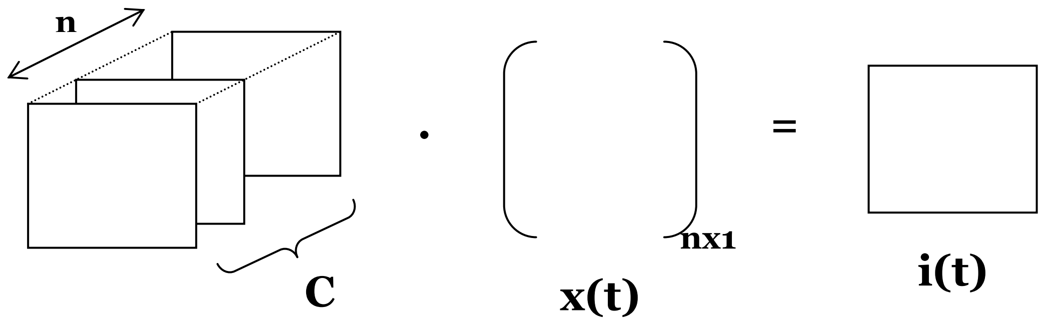


Figure 4-2: *Decomposition of $i(t)$ with n filters (see equation 4.2)*

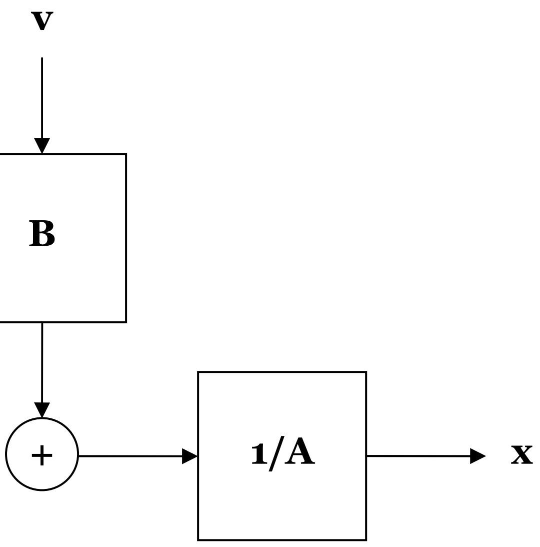


Figure 4-3: *State transitions of an ARMA model*

As described in [55], and shown in Figure-4.3, a stationary process $x(t)$ can be represented as $x(t) = R(q)v(t)$ where $v(t)$ is white noise. Here $R(q)$ is a rational (transfer) function.

$$\begin{aligned} R(q) &= \frac{B(q)}{A(q)} \\ B(q) &= 1 + b_1 q^{-1} + \cdots + b_{n_b} q^{-n_b} \\ A(q) &= 1 + a_1 q^{-1} + \cdots + a_{n_a} q^{-n_a} \end{aligned} \tag{4.3}$$

where q and q^{-1} are forward and backward shift operators. So that we may write

$$x(t) + a_1 x(t-1) + \cdots + a_{n_a} x(t-n_a) = v(t) + b_1 v(t-1) + \cdots + b_{n_b} v(t-n_b) \tag{4.4}$$

for equation (4.3). Such a representation of a stochastic process is known as an ARMA model. If $n_b = 0$, we have an auto-regressive (AR) model :

$$x(t) + a_1 x(t-1) + \cdots + a_{n_a} x(t-n_a) = v(t) \tag{4.5}$$

and if $n_a = 0$, we have a moving average (MA) model :

$$x(t) = v(t) + b_1 v(t-1) + \cdots + b_{n_b} v(t-n_b) \tag{4.6}$$

The proposed method in [1] uses $n_a = 1$ and $n_b = 1$, so in our case we have :

$$x(t) + a_1 x(t-1) = v(t) + b_1 v(t-1) \tag{4.7}$$

These relation is also represented in equation (4.8).

$$\underset{\mathbf{x(t+1)}}{\begin{pmatrix} \quad \\ \quad \\ \quad \end{pmatrix}_{\mathbf{nx1}}} = \underset{\mathbf{A}}{\begin{pmatrix} \quad \\ \quad \\ \quad \end{pmatrix}_{\mathbf{nxn}}} \cdot \underset{\mathbf{x(t)}}{\begin{pmatrix} \quad \\ \quad \\ \quad \end{pmatrix}_{\mathbf{nx1}}} + \underset{\mathbf{B}}{\begin{pmatrix} \quad \\ \quad \\ \quad \end{pmatrix}_{\mathbf{nxn_v}}} \cdot \underset{\mathbf{v(t)}}{\begin{pmatrix} \quad \\ \quad \\ \quad \end{pmatrix}_{\mathbf{n_vx1}}} \tag{4.8}$$

Equation (4.8) defines the state equation for ARMA model. Under above definitions the parts of an ARMA model is also represented in Figure-4.5. In Figure-4.5, we see that the source of the stochastical approach lies on moving average (MA) part, in other words the dependency of states to randomness creates undetermined results, while the auto-regression (AR) part of the structure make model dependent to past occurrences. So, we have a model which decides on the latest samples depending to its previous samples and updates the new ones under terms of randomness.

$$\underbrace{\underbrace{\mathbf{x(t+1) = \overbrace{A\,.\,x(t)}^{AR}}}_{\textbf{regression}} + \underbrace{\overbrace{\mathbf{B\,.\,v(t)}}^{MA}}_{\textbf{randomness}}}_{\textbf{ARMA Model}}$$

Figure 4-4: *Construction of an ARMA model : regression and randomness*

As we summarize, when we extend the equation (4.1) to image sequences, dynamic textures, we have a state-space model which can define the stochastical phenomenon of a dynamic texture. The equation (4.9) represents what we say the dynamic texture can be represented as a state-space model referring to [1] :

$$\begin{cases} X = AX + BV \\ Y = CX + W \end{cases} \tag{4.9}$$

4.1.2 Learning Phase : A Closed-Form Solution

When we examine the equation (4.9), we can see that the model defines a dynamic texture with only three parameters : A, B and C. So, the learning problem is a maximum likelihood learning similar to expectation maximization :

Given $\{y(1),\dots,y(\tau)\} = Y$, find

$$\hat{A},\hat{B},\hat{C},\hat{q}(\cdot) = \arg\max_{A,B,C,q} \log p(Y) \tag{4.10}$$

subject to equation (4.1) and $v(t) \overset{IID}{\sim} q$.

While Y is observable, solution is defined in [1] as follows :

Let $Y^\tau \doteq [y(1),\dots,y(\tau)] \in \Re^{mx\tau}$ with $\tau > n$, and similarly for $X^\tau \doteq [x(1),\dots,x(\tau)] \in \Re^{nx\tau}$, and $W^\tau \doteq [w(1),\dots,w(\tau)] \in \Re^{mx\tau}$ and notice that

$$Y^\tau = CX^\tau + W^\tau; \quad C \in \Re^{mxn}; C^T C = I \tag{4.11}$$

Now let $Y^\tau = U\Sigma V_s^T$; $U \in \Re^{mxn}$; $U^T U = I$; $V_s \in \Re^{\tau xn}$, $V_s^T V_s = I$ be the singular value decomposition (SVD) with $\Sigma = diag(\sigma_1,\dots,\sigma_n)$, and $\{\sigma_i\}$ be the singular values, and consider the problem of finding the best estimate of C in the sense of Frobenius: $\hat{C},\hat{X}(\tau) = \arg\min_{C,X^\tau} \|W^\tau\|_F$ subject to equation (4.11). It follows immediately from the fixed rank approximation property of the SVD that the unique solution is given by

$$\hat{C} = U\,,\ \hat{X}(\tau) = \Sigma V_s^T \tag{4.12}$$

$\hat{A}$ can be determined uniquely, again in the sense of Frobenius, by solving the following linear problem: $\hat{A} = \arg\min_A \|X^\tau - AX_0^{\tau-1}\|_F$, where $X_0^{\tau-1} \doteq [x(0),\dots,x(\tau-1)] \in \Re^{nx\tau}$ which is trivially done in closed-form using the state estimated from equation (4.12)

$$\hat{A} = X^\tau X_0^{\tau-1} \tag{4.13}$$

Following solution after equation (4.13) we get :

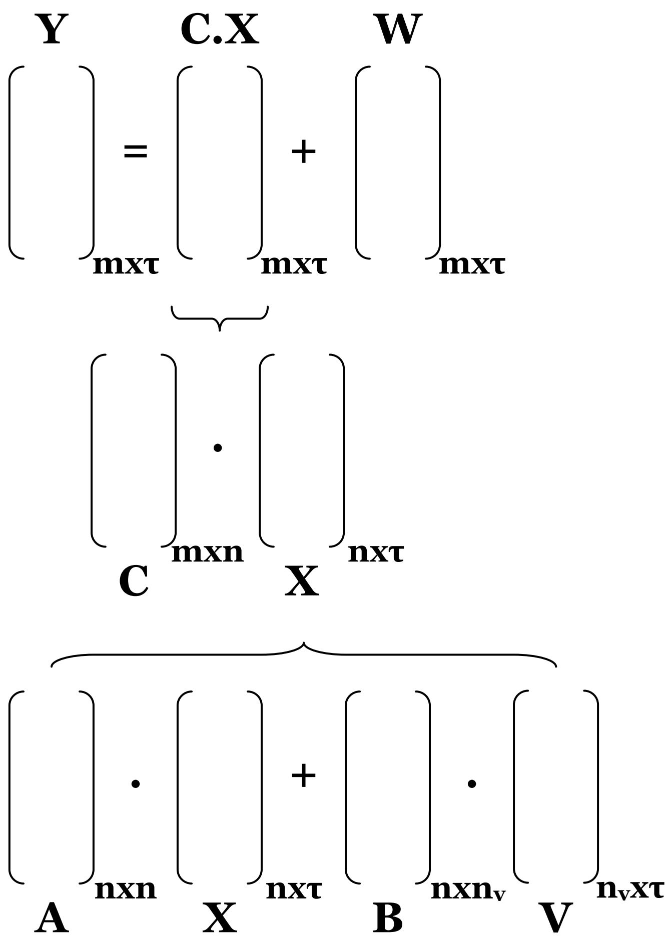
$$\hat{B}\hat{V}(\tau) = X^\tau - \hat{A}X_0^{\tau-1} \tag{4.14}$$

And now let $\hat{B}\hat{V}(\tau) = U_v \Sigma_v V_v{}^T$; $U_v \in \Re^{nxn_v}$; $U_v{}^T U_v = I$; $V_v \in \Re^{n_v x \tau}$, $V_v{}^T V_v = I$ be the singular value decomposition (SVD). It follows, as stated above, from the fixed rank approximation property of the SVD that the unique solution is given by

$$\hat{B} = U_v \Sigma_v \,,\ \hat{V}(\tau) = V_v{}^T \qquad (4.15)$$

where $V^\tau \doteq [v(1), \dots, v(\tau)] \in \Re^{n_v x \tau}$ and $B \in \Re^{nxn_v}$.

State-space relations of a dynamic texture in terms of matrices as an ARMA model is presented in equation (4.16) and the solution matrices of the system after SVD is given in equation (4.17).



(4.16)

$$\overset{\mathbf{Y}}{\begin{pmatrix} \quad \\ \\ \\ \end{pmatrix}}_{\mathbf{m x \tau}} = \overset{\mathbf{U}}{\begin{pmatrix} \quad \\ \\ \\ \end{pmatrix}}_{\mathbf{m x \tau}} \cdot \overset{\mathbf{\Sigma}}{\begin{pmatrix} \quad \\ \\ \\ \end{pmatrix}}_{\mathbf{\tau x \tau}} \cdot \overset{\mathbf{V^T}}{\begin{pmatrix} \quad \\ \\ \\ \end{pmatrix}}_{\mathbf{\tau x \tau}}$$

$$= \underbrace{\begin{pmatrix} \quad \\ \\ \\ \end{pmatrix}_{\mathbf{m x n}}}_{\mathbf{C}} \cdot \underbrace{\begin{pmatrix} \quad \\ \\ \\ \end{pmatrix}_{\mathbf{n x n}} \cdot \begin{pmatrix} \quad \\ \\ \\ \end{pmatrix}_{\mathbf{n x \tau}}}_{\mathbf{X}} \tag{4.17}$$

The solutions given above are all part of a system identification problem. Interested readers can study related parts of the books [54, 55, 56] for a better understanding of the solutions.

4.1.3 Clues Under Solution

What have we done by solving the problem using SVD ?

As explained in [12], there is a direct relation between principal component analysis (PCA) and SVD in the case where principal components are calculated from the covariance matrix. The eigenvalues of PCA are equivalent to σ_i^2 of SVD, which are proportional to the variances of the principal components. So, as the right singular vectors of SVD are the same as the principal components, taking $\hat{X}(\tau) = \Sigma V_s{}^T$ as also presented in equation (4.17), we take n biggest eigenvectors and corresponding eigenvectors for state representation. This yields, reduced dimension representation of the dynamic texture, plus showing trend of data in terms of reduced new dimensions. This property of the model also yields an advantage for using the model for synthesis and compression needs. Interested readers about PCA and SVD can find the subject in chapter 3 of this thesis or can refer to [11, 12, 54, 58].

4.2 Classification

The classification of dynamic textures which has modeled as an ARMA model as described in previous sections is a problem of calculating the distances between the parameters A, B and C, whom they constitute and describe the dynamic texture, between two models, M_1 and M_2. The solution is given in [1] referring to [2] and [3]. In [2], a notion of principal angles and their corresponding principal directions between two linear autoregressive moving average (ARMA) models is defined and their relation to the metric for ARMA models defined by Martin [3] is shown.

4.2.1 Subspace Angles and Martin Distance Between Models

Let $A \in R^{mxp}$ and $B \in R^{mxp}$ be two matrices with full column rank. The principal angles $\theta_k \in \left[0, \frac{\pi}{2}\right]$ between $range(A)$ and $range(B)$ are defined as

$$\cos(\theta_k) = \max_{\substack{x \in R^p \\ y \in R^q}} \frac{\left|x^T A^T B y\right|}{\|Ax\|_2 \|By\|_2}, \text{ for } k = 1,2,\ldots,\min(p,q)$$

Subspace angles are the largest of these angles. A closed form solution is presented in [2]. Notice that, in our case, we seek subspace angles between models M_1 and M_2.

For the sake of simplicity, assuming that we have AR models, that we will deal only with parameters A and C. However this assumption is done, in [2] it is also shown that the resulting equality holds for ARMA models. So, currently assuming we have AR models, the models will be represented as $M_1 \doteq (A_1, C_1)$ and $M_2 \doteq (A_2, C_2)$. And the observability matrix for models will be $O_\infty(M_i) \doteq \left[C_i^T \quad A_i^T C_i^T \quad \cdots \quad (A_i^T)^n C_i^T \quad \cdots\right]$.

The subspace angles between M_1 and M_2 is defined as the principal angles between the ranges of their infinite observability matrices:

$$[M_1 \lhd M_2] = [O_\infty(M_1) \lhd O_\infty(M_2)] \tag{4.18}$$

In [3], the distance between stable AR models M_1 and M_2 is defined as:

$$d(M_1, M_2) = \sqrt{\sum_{n=0}^{\infty} n \left| c_1(n) - c_2(n) \right|^2} \tag{4.19}$$

where $c_1(n)$ and $c_2(n)$ are cepstrum coefficients of M_1 and M_2, respectively. (The cepstrum of a discrete-time process is the inverse Fourier transform of the logarithm of the power spectrum of the discrete-time process.)

As presented in [2], Martin [3] subsequently shows that for stable AR models M_1 with order n_1 and poles α_i and M_2 with order n_2 and poles β_i the following equality holds

$$d(M_1, M_2)^2 = \ln \frac{\prod_i^{n_1} \prod_j^{n_2} \left| 1 - \overline{\alpha}_i \beta_j \right|^2}{\prod_{i,j}^{n_1} (1 - \overline{\alpha}_i \alpha_j) \prod_{i,j}^{n_2} (1 - \overline{\beta}_i \beta_j)} \tag{4.20}$$

This equality basically follows from an expression that relates the cepstrum coefficients of an AR model to its poles, which can be found in [20].

Observe that if M_1 and M_2 are two first order stable AR models, their distance equals

$d(M_1, M_2)^2 = \ln \frac{(1-\alpha\beta)^2}{(1-\alpha^2)(1-\beta^2)} = \ln \frac{1}{\cos^2 \theta}$, where θ is the subspace angle between models M_1 and M_2. For n^{th} order models, the distance is defined as:

$$d(M_1, M_2)^2 = \ln \prod_{i=1}^{n} \frac{1}{\cos^2 \theta_i} \tag{4.21}$$

The proof can be found in [2].

Finally, the Martin distance between two models of order n is defined as :

$$d_m(M_1,M_2)^2 = \ln \frac{\prod_{i,j}^{n} \left|1-\overline{\alpha}_i \beta_j\right|^2}{\prod_{i,j}^{n} (1-\overline{\alpha}_i \alpha_j)(1-\overline{\beta}_i \beta_j)} \tag{4.22}$$

where α, β are the poles and $\overline{\alpha}$, $\overline{\beta}$ means complex conjugates of poles of M_1 and M_2, respectively.

In [2], it is declared that the equation (4.22) is also valid for ARMA models.

The poles of an ARMA model is the eigenvalues of A which is the state transition matrix (see equation (4.3) and books [55, 56] for details).

In our implementation, equation (4.22) is used for calculating distances between models.

CHAPTER 5

DETERMINISTIC APPROACH

In this chapter, the deterministic solution to dynamic texture classification problem is defined. The solution depends on the paper by Peteri [4]. This paper is selected because of its reported success and also being one of the latest works on dynamic texture classification in literature. However we focused on classification of dynamic textures, we have also worked on texture regularity measuring which is one of the main feature extraction mechanisms of the deterministic method. Interested readers can also refer to papers [5, 6, 7, 14, 15, 16, 23, 26, 27, 28] for constituting the best knowledge about the subject.

This method aims to extract spatial and temporal features of a dynamic texture using a texture regularity measure and normal flow. Texture regularity is calculated referring to [5] and normal flow is calculated referring to [6].

5.1 Normal Flow

As stated in [4], normal flow contains both temporal and structural information of dynamic textures : temporal information is related to moving edges, while spatial information is linked to the edge gradient vectors. So, it is suitable for dynamic textures. The method is also fast for computing. However, the weakness lies on its sensitivity to noise. This aspect of method is improved by smoothing the image or applying a threshold on spatial gradients. Features using normal flow information is obtained from motion vectors of each individual pixel which are calculated via the well known optical flow calculation methods. Features that are used for classification are explained in section 5.4 of this chapter. For more information about optical flow, interested readers can find the subject in chapter 3 of this thesis or can refer to [5, 26, 27].

5.2 Texture Regularity

In recent years, this subject is under investigation by different research groups. The work done by Chetverikov in [5] is the most succesful one that I have found in literature. The terms of success for texture regularity depends on qualitative evaluation. We have used the method proposed in [4] referring to [5] and [23]. We have also add an improvement to the method. The results that we had obtained are better than the proposed method in [5]. The comparison of our results with the ones in [5] are shown in chapter 6. You can also find our results with entire Brodatz album which became the standart evaluation database about these subject.

5.2.1 Regularity Measure

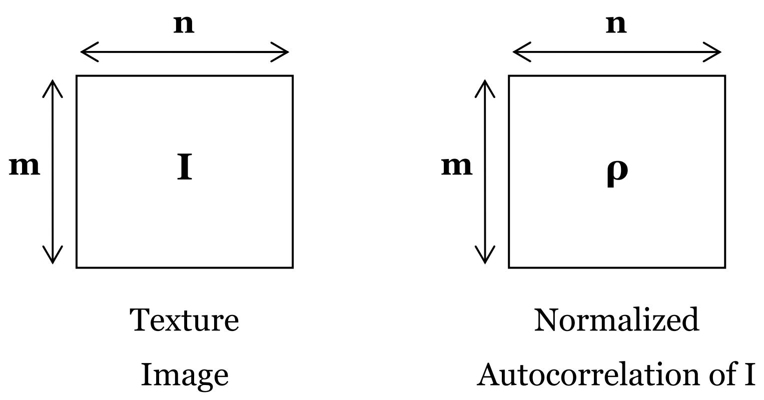
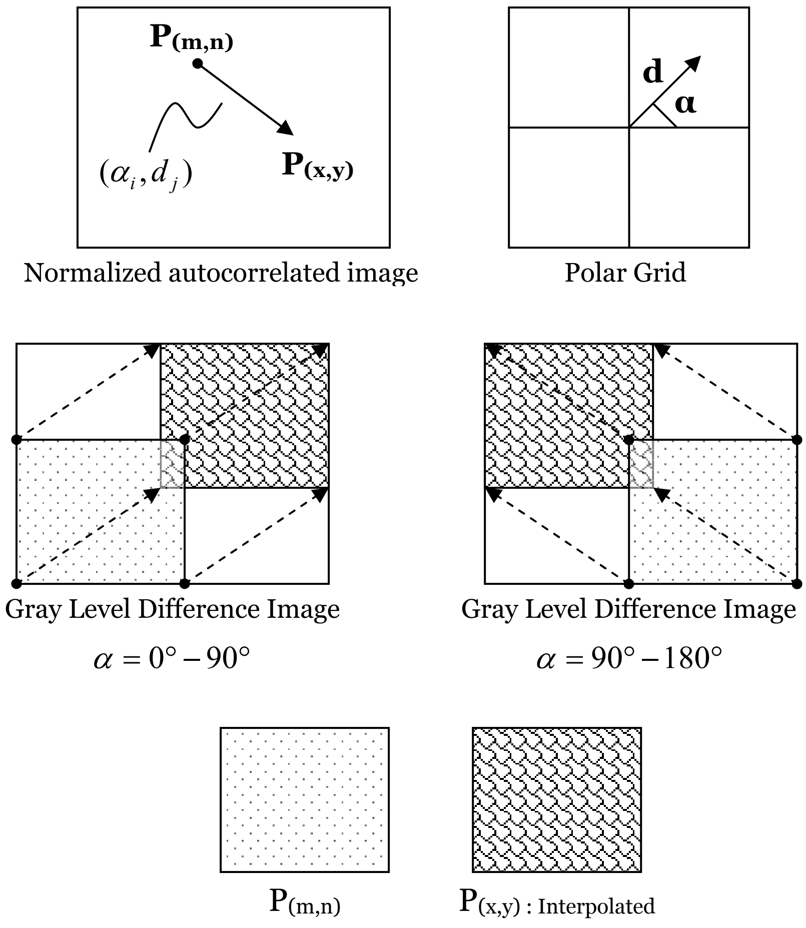


Figure 5-1: *Normalized autocorrelation of texture image*

Regularity measure depends on seeking similarities and measuring periodicity of these similarities in a texture image. For this purpose, the autocorrelation of the original texture image is calculated via the FFT using the well-known relation between the correlation function and the Fourier transform. Autocorrelation is then normalized by spreading the results between minimum and maximum gray levels. The result of this process emphasizes the similarities in texture image. For detecting periodicity of these similarities, gray level differences are calculated in normalized autocorrelation of the original texture image. This is done by surveying in image with vectors whose distances range from $d = \Delta d$ to $d_{\max}$ and angles range from $\alpha = \Delta\alpha$ to 2π, where Δd is the increment size of distances and $\Delta\alpha$ is the increment size of angles. Gray level differences are calculated simply using the pixel differences of which the vector is showing at its starting and finishing points. While we search the vector in image using the pixels at its start point, the finishing point must be calculated using four pixels neighboring almost the finishing point that the vector points. In simple words, calculate a four pixel interpolated image and take difference image in range of the vector. See Figure-5.2 for details.



P(m,n)
(αi, dj)
P(x,y)
Normalized autocorrelated image
d
α
Polar Grid
Gray Level Difference Image
α = 0° − 90°
Gray Level Difference Image
α = 90° − 180°
P(m,n)
P(x,y) : Interpolated

***Figure 5-2 :** Calculation hints for gray level differences*

Using the gray level difference image, the gray level difference histogram is calculated. Then, the polar grid is calculated using the histogram weighted mean of the gray level difference image.

For all (α_i, d_j) ;

$$mean(i,j) = \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}} kH(k;\alpha,d) \tag{5.1}$$

where $k = |\rho(m,n) - \rho(x,y)|$, $x = n + d\cos\alpha$, $y = m - d\sin\alpha$, $k_{\max}$: number of gray levels – 1, $H(k;\alpha,d)$: histogram of gray level difference image, $\alpha_i = i.\Delta\alpha$, $d_j = j.\Delta d$, $N_\alpha = 2\pi / \Delta\alpha$, $N_d = d_{\max} / \Delta d + 1$. N_α and N_d are two basic parameters of this calculation. It is assumed that $d_{\max}$ is greater than at least two periods of the pattern. Our parameter selections and results are shown in chapter 6.

The mean function calculated for the polar grid is then normalized with its maximum, so $\rho_{pol}(i,j)$ is obtained where it ranges now 0 to 1. Finally, the inverse of $\rho_{pol}(i,j)$ is calculated which is given as

$$M_{pol}(i,j) = 1 - \rho_{pol}(i,j) \tag{5.2}$$

where $0 \le M_{pol}(i,j) \le 1$ is called the polar interaction map. A row of the polar interaction map is called the contrast function. Regularity is the result of interpretation of this function. Notice that, rows of contrast function are the increasing angles α_i and the columns of the contrast function are the increasing displacements d_j. The plot of α_i according to d_j gives us the contrast curve. A regular texture has a contrast curve with deep and periodic minima. See Figure-5.3 for visual explanation of the measure.

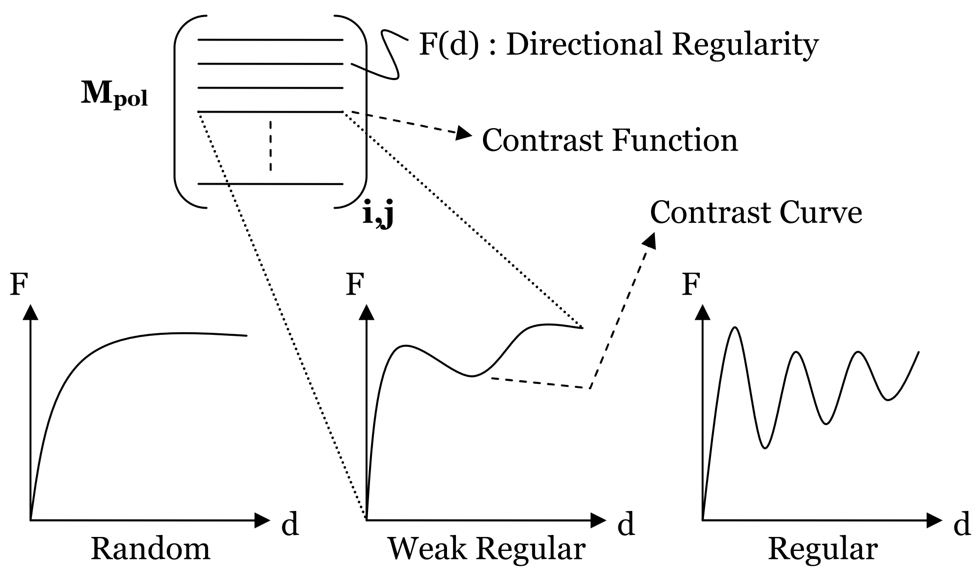


Figure 5-3 : *Contrast curves and their relation between regularity*

Using contrast curves the regularity measure is calculated step by step as follows as stated in [4] :

1. The contrast function is median filtered with a size of three. This filtering smoothes the function : $F(d) = median(F_i(d),3)$.
2. Find extrema of $F(d)$.
3. Select two lowest minimums : $(d_1, F_1), (d_2, F_2)$, $d_1 < d_2$.
4. Take the lowest minimum of $F_i(d)$, F_{am} : get rid of filtered value, use the original minimum which is at a ± two point vicinity of $d(\min(F_1, F_2))$.
5. $REG_{\text{int}} = 1 - F_{am}$
6. Calculate periodicity in terms of distances : $REG_{pos} = 1 - \left|\frac{d_2 - 2d_1}{d_2}\right|$
7. $REG(i) = (REG_{\text{int}} . REG_{pos})^p$: where p is a parameter, selected as 2.
8. $MAXREG = \max_i(REG(i))$ (5.3)

At step 6, also consider a special case which the third minimum is occurred between two lowest minimums. If special case is detected then calculate also $REG_{pos} = 1 - \left| \frac{d_2 - 3d_1}{d_2} \right|$. Use the maximum of these two values. See Figure-5.4 for details.

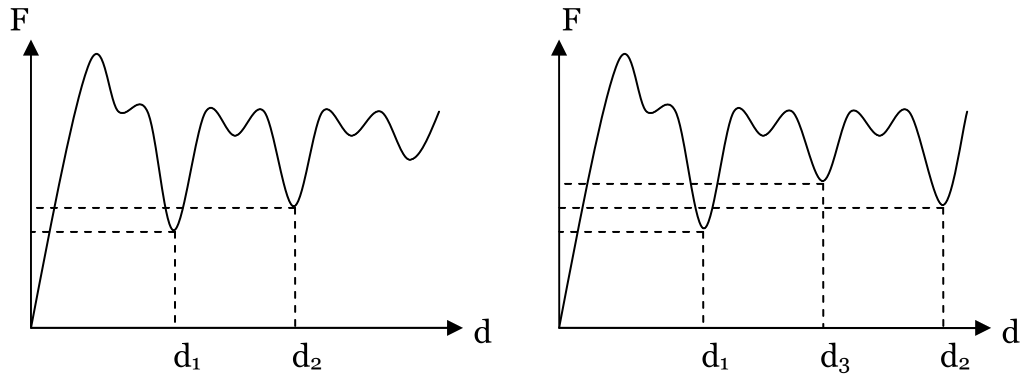


Figure 5-4 : *Special case for* REG_{pos}

5.2.2 Improvements for Regularity Measure

During our evaluations we see that the method proposed in [4], as explained in section 5.2.1, has some weaknesses while interpreting the contrast curves for textural images. In their latest work in [23], they have also proposed a new method for REG_{int}. The new method calculates REG_{int} as follows :

- Calculate the differences between each maximums and minimums. Select the largest amplitude $F_{\max} - F_{\min}$. Then, $REG_{\text{int}} = 1 - \frac{F_{\min}}{F_{\max}}$.

However, this addition solves the previous weakness, it is not exactly enough for a true determination of texture regularity. The positional evaluation of the curve also has weaknesses and we have proposed some new calculations for solving these.

5.2.2.1 Improvement 1 : positional award

When contrast curves of textural images are investigated it can be noticed that most of time the positional distribution of periodicities can not be well represented with two lowest minimums. In case of these, a very regular image has also score bad results, in other words, it is punished unfairly. But, this weakness of the method can be removed by taking a third minimum. Looking periodicity between three minimums arise better results. However, this improvement is only take in considered if all three lowest minimums are sorted in ascending order in terms of d. So, this calculation will be an award for differentiating regular images. Figure-5.5 shows details.

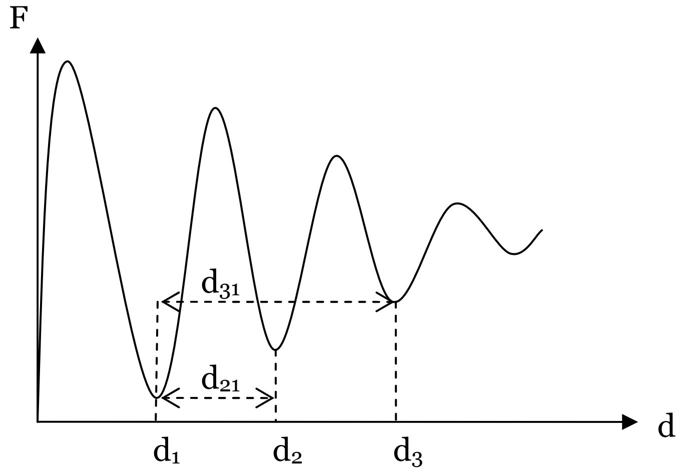


Figure 5-5 : *New proposed positional regularity measure*

Our proposed method calculates positional award as :

$$AWARD_{pos} = 1 - \left| \frac{d_{31} - 2.d_{21}}{d_{31}} \right| \tag{5.4}$$

where $d_{31} = d_3 - d_1$ and $d_{21} = d_2 - d_1$. Then the method uses :

$$REG_{pos} = \max(REG_{pos}, AWARD_{pos}) \tag{5.5}$$

5.2.2.2 Improvement 2 : value regularity

The contrast curve in Figure-5.5 is a good example for high regularity. In regular textures, the contrast curve also has periodicity through F. Our improvement method adds this property to regularity measure. Figure-5.6 shows details of proposed method.

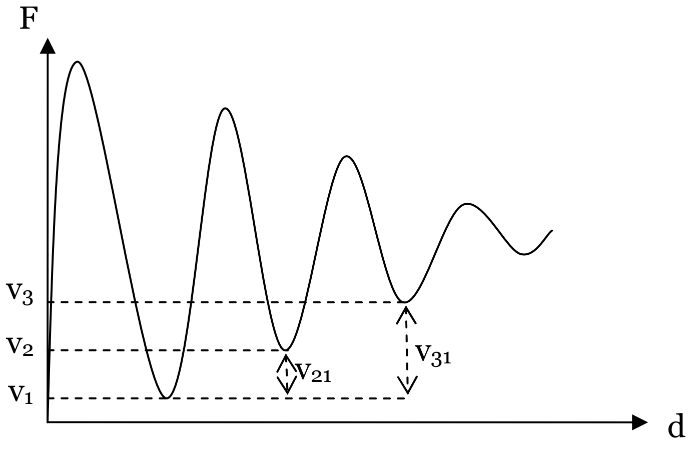


Figure 5-6 : *New proposed value regularity measure*

Our proposed method calculates value regularity as :

$$REG_{val} = 1 - \left| \frac{v_{31} - 2.v_{21}}{v_{31}} \right| \tag{5.6}$$

where $v_{31} = v_3 - v_1$ and $v_{21} = v_2 - v_1$. If there exists only two minimums then the calculation is :

$$REG_{val} = 1 - \left| \frac{v_2 - 2.v_1}{v_2} \right| \tag{5.7}$$

The new calculated measure REG_{val} effects the total regularity measure as follows :

$$REG(i) = (REG_{\text{int}}.REG_{pos}.REG_{val})^p \tag{5.8}$$

where $p = 2$. However this additional measure reduces the resulting regularity score of textures even it is high regular, we use an additional decision criteria based on the idea that : “If a texture has regular characteristic, then it must score high in all of the three regularity measures”. So, a threshold is applied and the final regularity becomes :

If (all regularity measures, (REG_{int}, REG_{pos}, REG_{val}) > t) then $p = 1$, where t is a threshold of our decision criteria which is selected empirically as $t = 0{,}8$.

5.3 Features

Deterministic approach to dynamic texture classification proposed in [4], selects the features according to normal flow and texture regularity as follows.

1. **Divergence :** Average divergence effect over the whole image sequence (dynamic texture) which shows the aim of normal flow field to scaling.
2. **Curl :** Average curl effect over the whole image sequence (dynamic texture) which shows the aim of normal flow field to rotation.
3. **Peakness :** It is defined as the average flow magnitude divided by its standard deviation.
4. **Orientation :** This is the orientation homogeneity of the normal flow field : $\phi = \frac{\left\| \sum_{i \in \Omega} \upsilon_N^i \right\|}{\sum_{i \in \Omega} \left\| \upsilon_N^i \right\|} \in [0,1]$ where υ_N^i is the normal flow vector at i and Ω is the set of points with non-zero normal flow vectors. ϕ reflects the flow homogeneity of the dynamic texture compared to its mean orientation. A detailed description of its meaning can be found in [7].

5. **Mean of regularity :** Texture regularity of each image is calculated using equation (5.3), which we called $MAXREG(t)$. The mean of $MAXREG(t)$ gives the temporal regularity of the dynamic texture.

6. **Variance of regularity :** The variance of temporal regularity.

The first four features depends on the normal flow field, while the last two are obtained from texture regularity.

5.4 Classification

The classification of the dynamic textures is simply done by calculating the weighted distances between the sample set and the classes which consist of sets. In this classification, all features except the fourth, orientation, feature have weights equal to 1. Orientation feature is weighted with 3.

$$D(s,c) = \sum_{i=1}^{6} w_i \left\| F_i^{(s)} - \overline{F}_i^{(c)} \right\| \tag{5.9}$$

The sample set s is classified as belonging to the nearest class n : $D(s,n) < D(s,c)$ for all $c \neq n$.

CHAPTER 6

IMPLEMENTATION & RESULTS

In this chapter, details of implementation and results are shown on subjects:

1. Construction of database
2. Texture regularity measuring depending on deterministic approach
3. Dynamic texture classification via both approaches
4. Dynamic texture synthesis depending on stochastic approach

The word database, here, is used for collection of dynamic textures or textures according to the case. Notice that, the constructed database which is explained in section 6.2 is simply a library of dynamic textures which we have created from a known database in literature.

6.1 Programming

All of our works except database manipulations are implemented using Matlab, version 7. Database manipulation is realized via C++. Borland C++ Builder, version 6 is used for design and compilation of the C++ code. Dynamic texture classification code is programmed with a graphical user interface of Matlab which supplies a very interactive and flexible usage. A view of our program is presented in Figure-6.16 and Figure-6.17 which are located at the end of this chapter.

6.2 Database

Dynamic texture database used in this thesis is downloaded from :

http://vismod.media.mit.edu/pub/szummer/temporal-texture/raw/

The MIT Temporal Texture database consists of 32 different dynamic textures. Files are in raw format each having its own description file.

Dynamic texture files contain image texture sequences ranging from 70 frames to 150 frames. Frame resolutions vary from (90 x 150) to (256 x 256) totaly at 7 different sizes. Also, description files are not in same readable format.

Under this circumstances, a database manipulation, to use for implementation, is needed. For this purpose, a program is designed to achieve a proper database. First of all, description files are re-organized to include information of width, height and number of total frames. For easy to use, all data files are named as “data.ris” and description files as “desc.txt”.

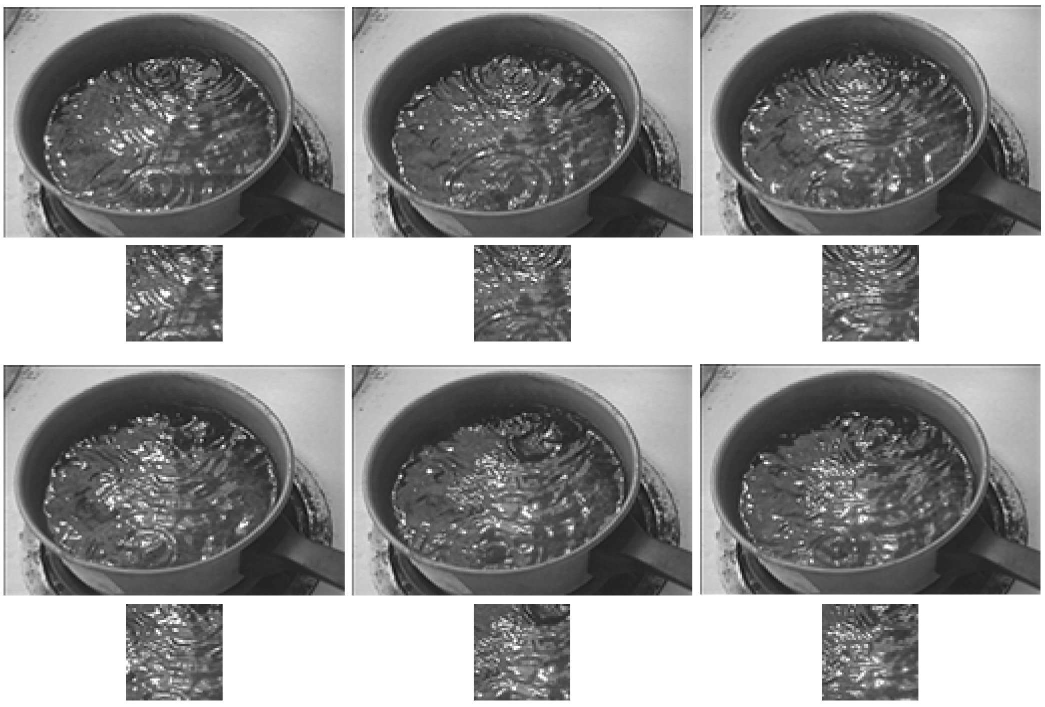


Figure 6-1: *Six frames of light boiling water and the belonging sub-sampled frames with a window of size (48 x 48) are shown up and down, respectively.*

A new database is created using the MIT Temporal Texture database, say it the database is enlarged and organized. The most significant window, i.e. flickering fire, boiling water, etc., at size (48 x 48) is focused on the image sequences and 60 frames of each are sampled. By this method, the database is enlarged to a database having 384 different dynamic textures. See Figure-6.1 and Figure-6.2.

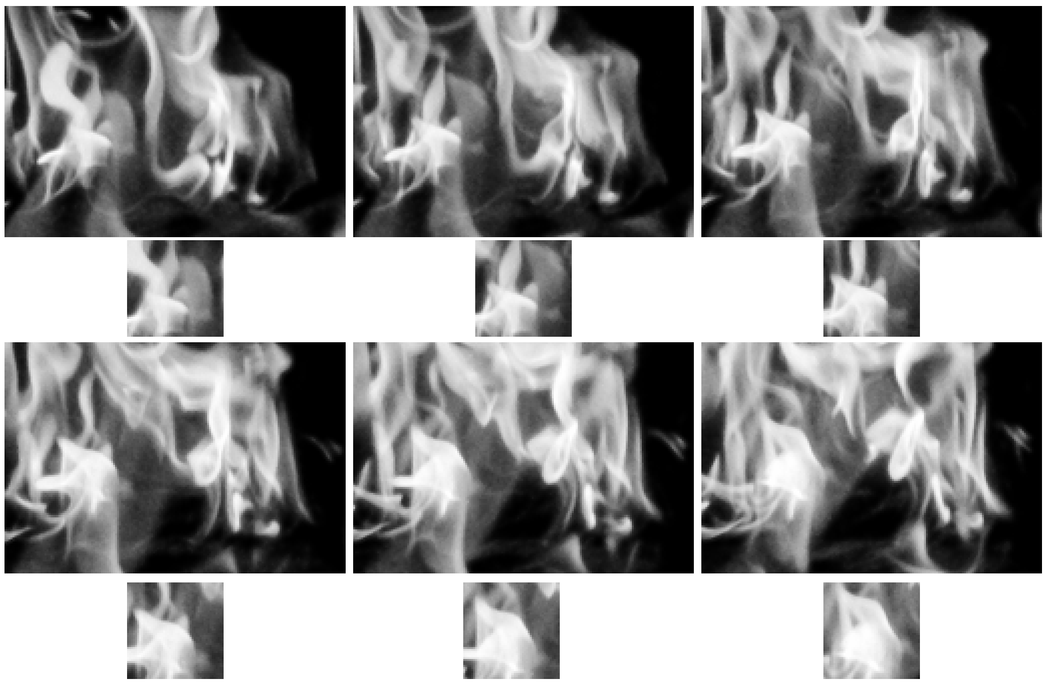


Figure 6-2: *Six frames of flickering fire and the belonging sub-sampled frames with a window of size (48 x 48) are shown up and down, respectively.*

6.3 Texture Regularity

Measuring the regularity of a texture is under investigation by different research groups. However it is possible to evaluate most of the results of the proposed methods of these research groups, the terms of success depends on qualitative evaluations. Because of this situation, Brodatz's album of textures became a standart evalution database for this subject. The Brodatz texture database can be downloadable at web address :

http://www.ux.his.no/~tranden/brodatz.html

The results of our method, which has improvements to the proposed methods in [4], [5] and [23] as described in sections 5.2.2.1 and 5.2.2.2 of this thesis, on Brodatz texture database are shown in this section. We have also compared our results with the ones given in [5]. The textures belong to this comparison can also be downloadable at web address :

http://visual.ipan.sztaki.hu/regulweb/node5.html

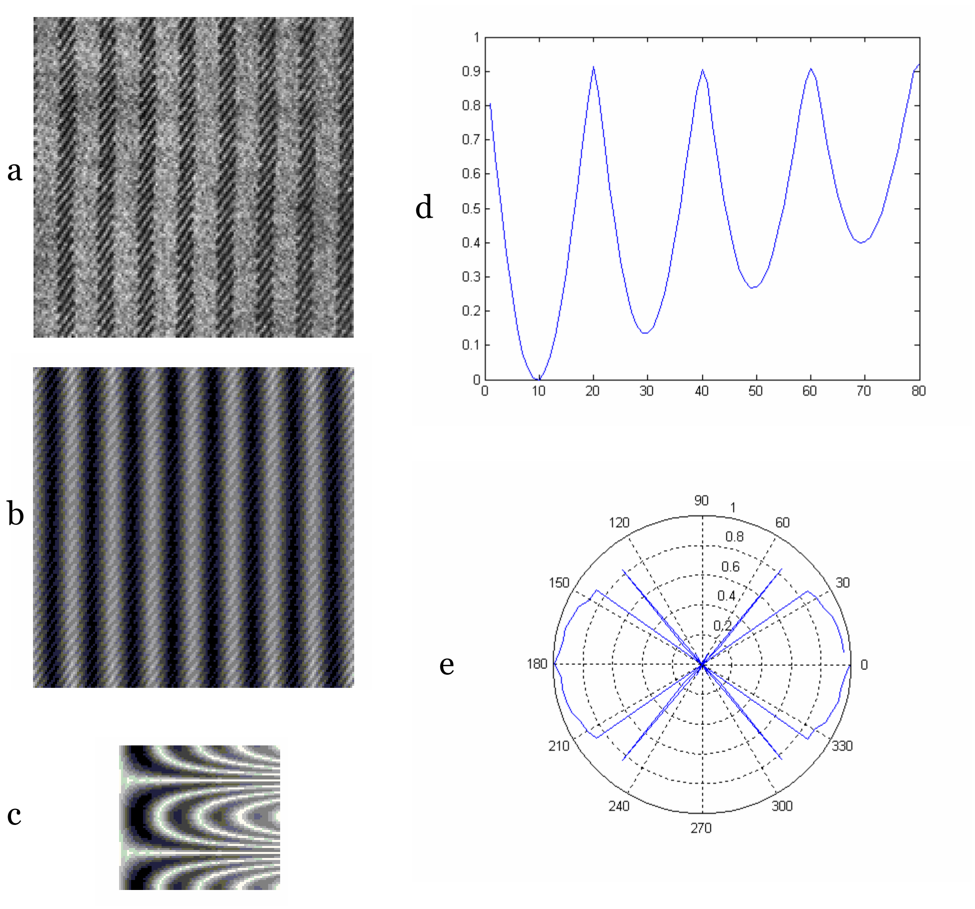


Figure 6-3: *a) original image. b) normalized autocorrelation of a. c) polar interaction map of b. d) contrast curve of the angle which has biggest score in c. e) polar plot of scores of all angles in c.*

The steps of measuring texture regularity, advantages and drawbacks of our method will be discussed on following figures starting from Figure-6.3. In figures starting from Figure-6.3 to Figure-6.7 except Figure-6.5, an original texture image, the normalized autocorrelation of the image, belonging polar interaction map, contrast curve at where the biggest score occurred and the polar plot of all scores at all angles are shown.

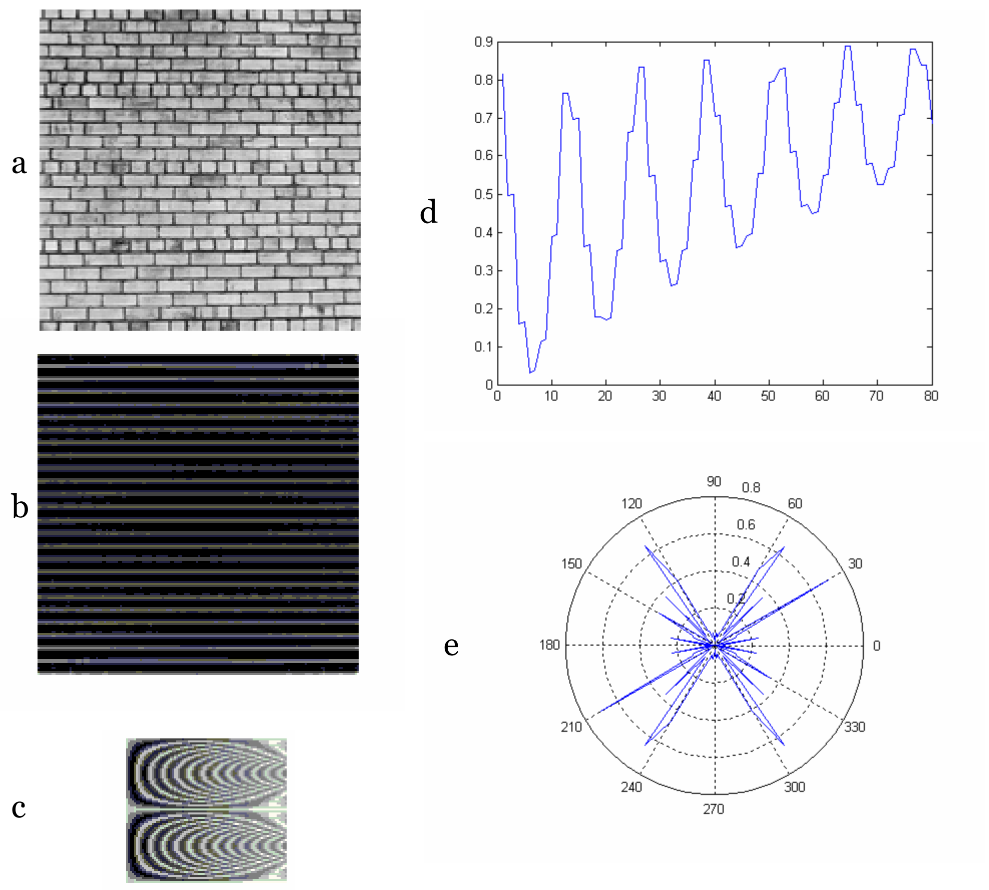


Figure 6-4: *a) original image. b) normalized autocorrelation of a. c) polar interaction map of b. d) contrast curve of the angle which has biggest score in c. e) polar plot of scores of all angles in c.*

A very regular texture is shown in Figure-6.3 which has a texture regularity score of 0,99174. The calculation of polar interaction map from normalized autocorrelation of the original image depends on three basic parameters which are maximum size of, increment step size of and angular increment step size of the displacement vector that we look up the gray level differences in the normalized autocorrelation of the original image by referencing this vector. We have chosen these parameters as follows: half of the image's longest axis as maximum size of the vector, 1 for increment step size and 5° for angular increment step size of the vector.

The texture in Figure-6.4 has a texture regularity score of 0,70345. The score means that the texture is regular. Regularity score of a texture can change from 0 to 1 where 0 means random texture and 1 means a very regular texture. As the details on calculations of the texture regularity are given in chapter 5 of this thesis, some of the drawbacks of these calculations will be exposed here. Notice that regularity score is nothing else than interpreting the contrast curves. As proposed in [5], the curves are median filtered with a size of three. Then, the interpretation is done on these filtered curves. But as it can be seen in Figure-6.4, the maximum score occurs at angle 210° whereas we expect it should be on angles 90° and 270°. Polar interaction map shows that nothing wrong with the data calculated. As we examine the normal and filtered plots at angle 90°, we can understand where the wrong is. These plots are shown in Figure-6.5.

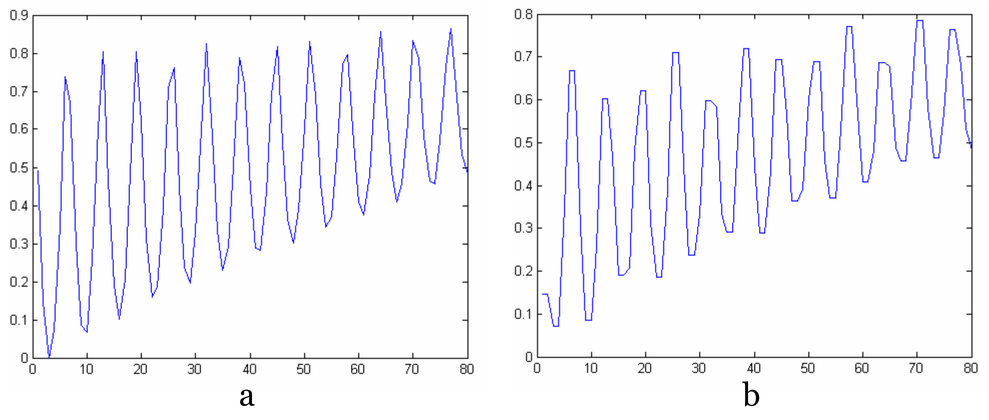


Figure 6-5: *a) contrast curve at 90° of c in previous figure. b) median filtered contrast curve of a.*

In Figure-6.5, we can see that the minimums and maximums are corrupted too much and the final scoring of texture at this angle drops down sharply. This situation occurs on texture images where the regular patterns are too close to each other. An example of this can be seen also in Figure-6.6.

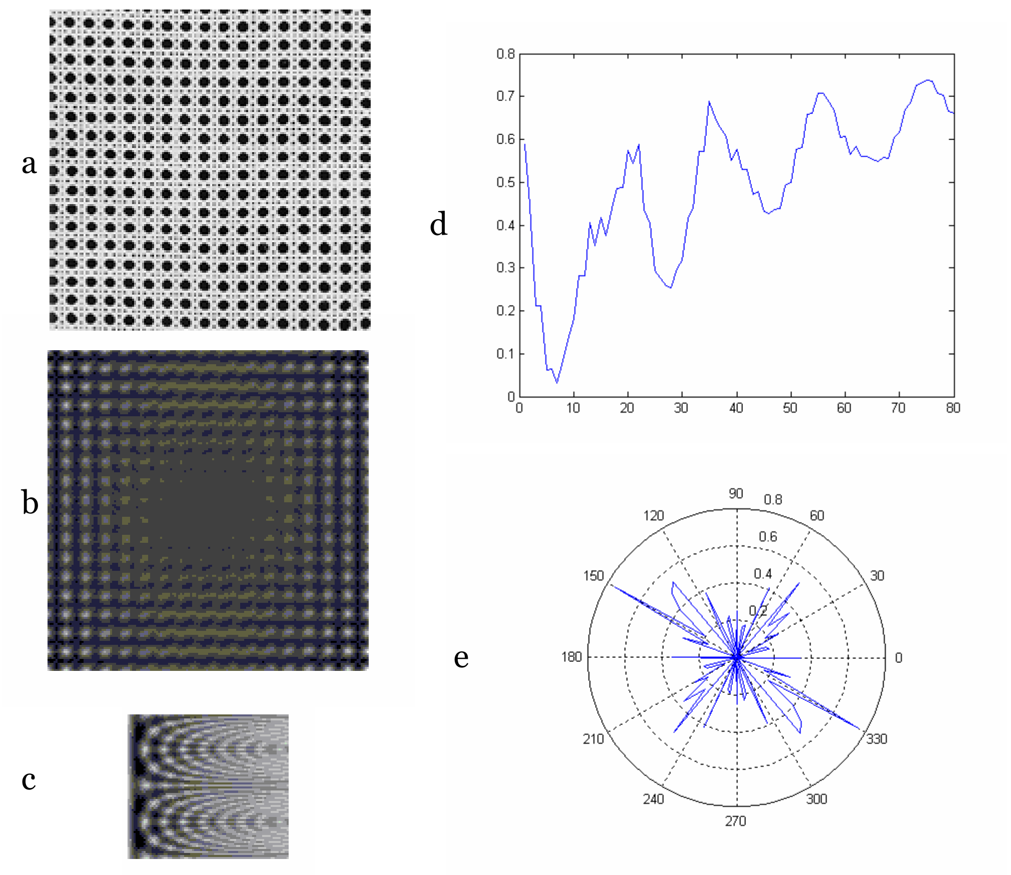


Figure 6-6: *a) original image. b) normalized autocorrelation of a. c) polar interaction map of b. d) contrast curve of the angle which has biggest score in c. e) polar plot of scores of all angles in c.*

In Figure-6.6, texture regularity score shows 0,75753 whereas it must be expected as almost 1. So far here we have examined some regular textures. A texture which has random regularity and its plots are shown in Figure-6.7.

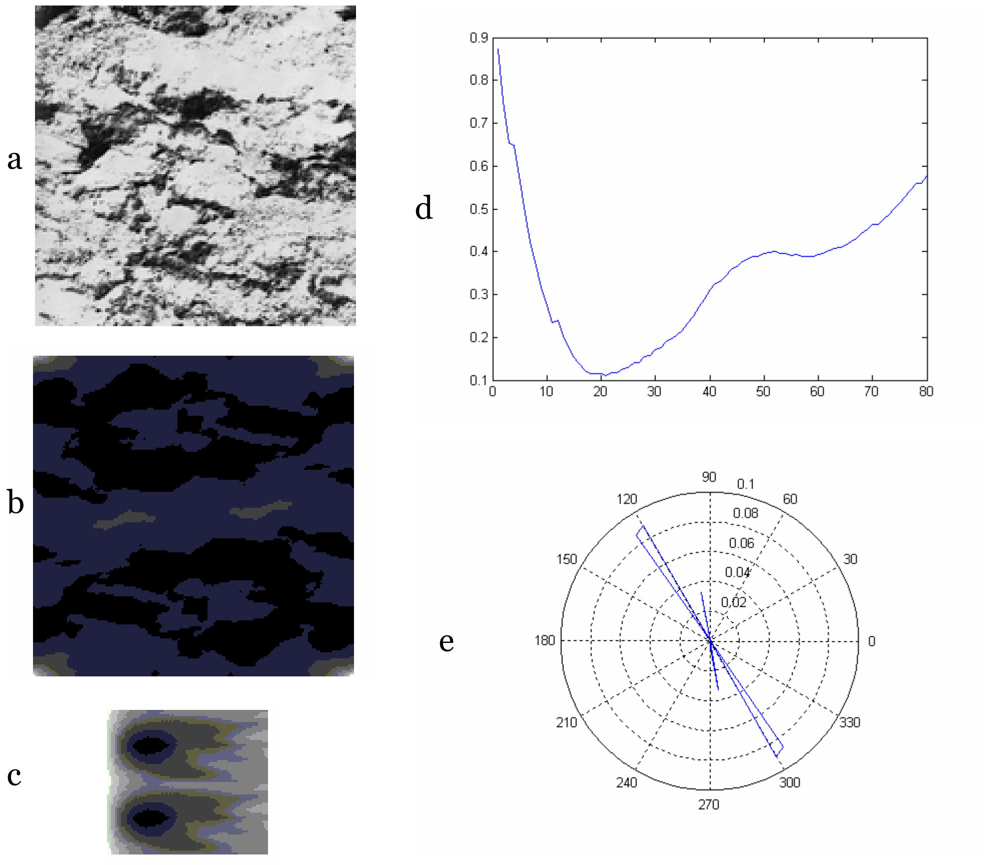


Figure 6-7: *a) original image. b) normalized autocorrelation of a. c) polar interaction map of b. d) contrast curve of the angle which has biggest score in c. e) polar plot of scores of all angles in c.*

Before giving our regularity measure results on Brodatz texture database, a comparison of the results given in [5] and ours is shown in Table-6.1.

The Brodatz texture database has texture images with size 640x640 in pixels. Because of computation speed, this database is sub-sampled and the sizes are dropped down to 160x160 in pixels. Texture regularity measuring is then calculated on these sub-sampled images. However the database has more than hundred images, all of the results are shown in Table-6.2.

Table 6-1: Comparison of results; rows 1: given in [5], rows 2: ours.

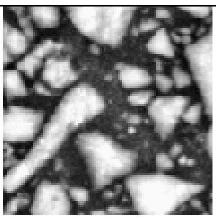
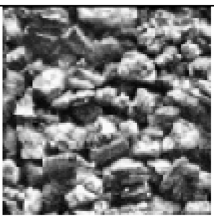
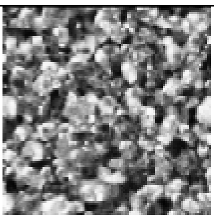
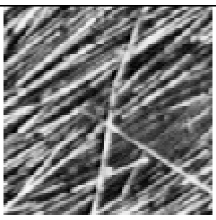
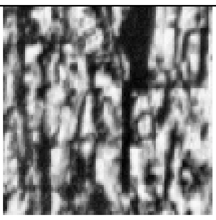
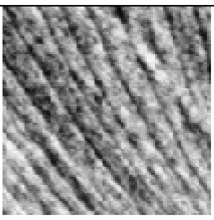
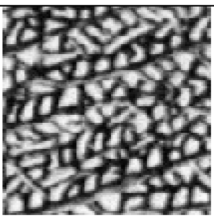
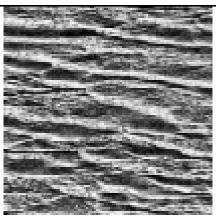
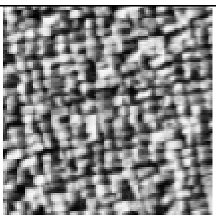
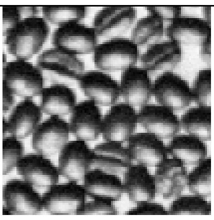
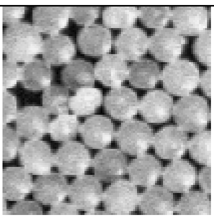
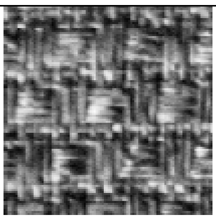
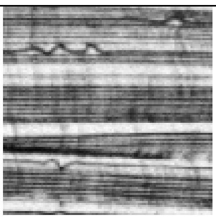
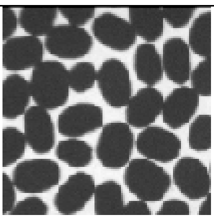
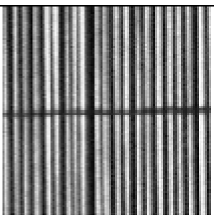
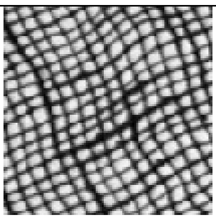
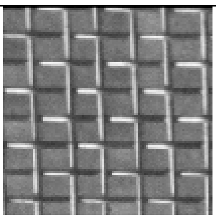
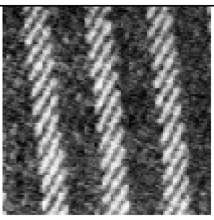
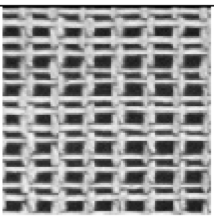
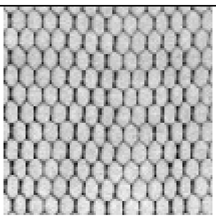
			
0.00	0.07	0.11	0.18
0,072695	0,29117	0,22543	0,32623
			
0.22	0.25	0.29	0.32
0,37577	0,5199	0,79676	0,90523
			
0.36	0.39	0.5	0.54
0,81625	0,28317	0,92301	0,14813
			
0.58	0.61	0.71	0.75
0,060503	0,4899	0,7894	0,88071
			
0.82	0.87	0.95	1.00
0,6668	0,001617	0,91249	0,93393

Table 6-2: Regularity results we have obtained on Brodatz texture database.

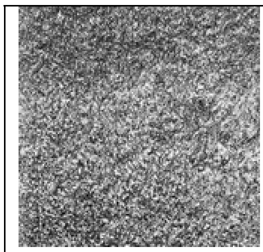
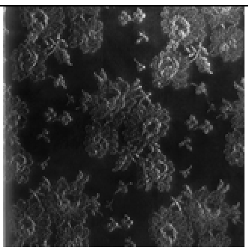
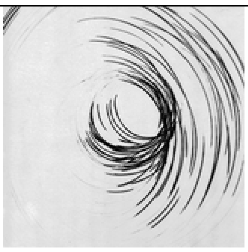
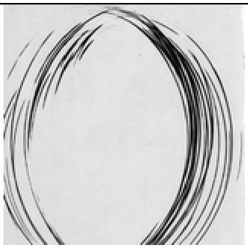
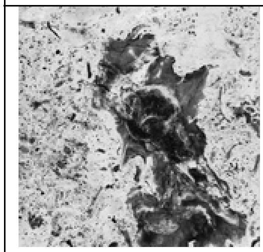
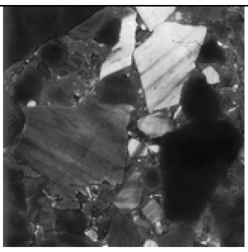
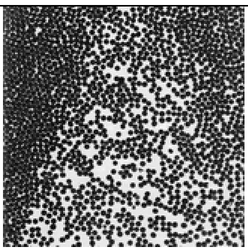
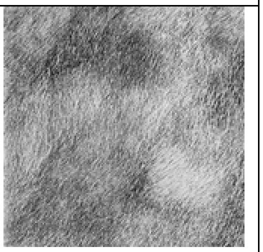
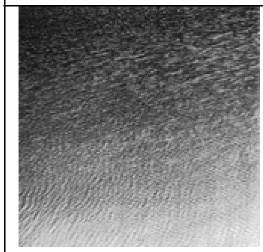
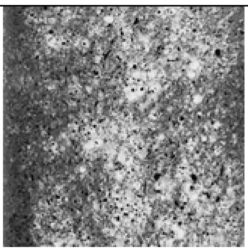
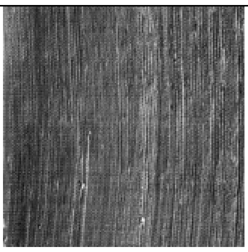
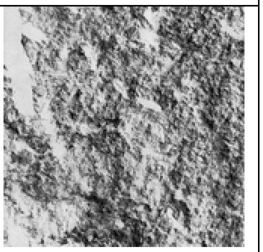
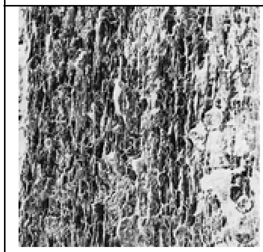
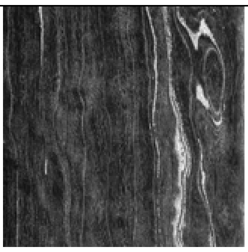
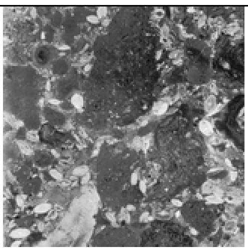
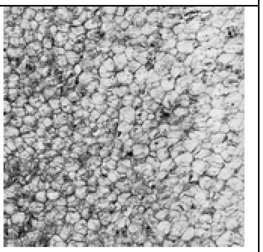
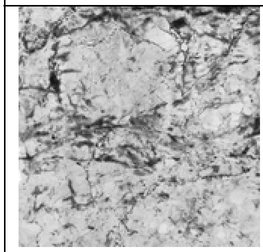
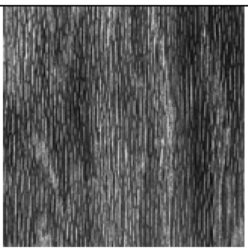
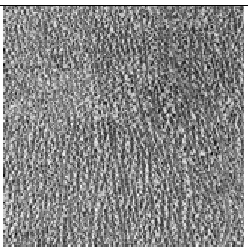
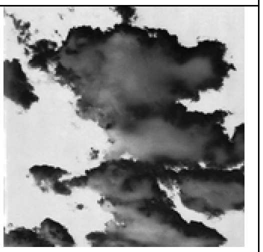
			
0	0	0	0
			
0	0	0	0
			
6,71E-06	9,54E-06	3,74E-05	6,21E-05
			
9,78E-05	0,000293	0,00030418	0,000689
			
0,00075406	0,0013747	0,0014088	0,001831

Table 6-2 (continued)

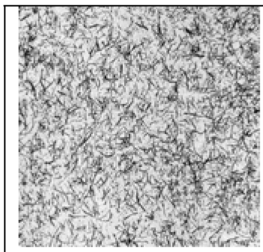
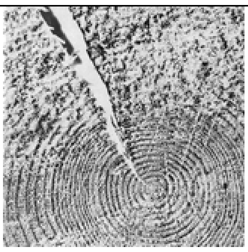
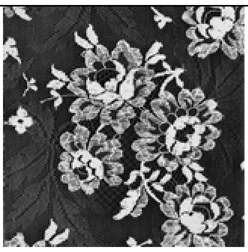
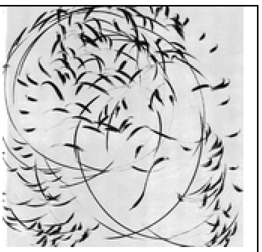
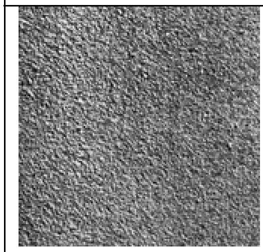
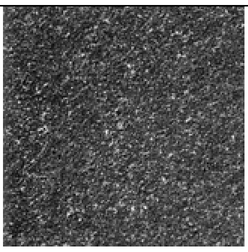
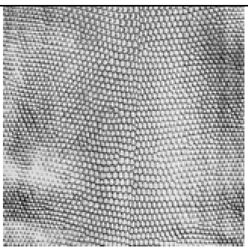
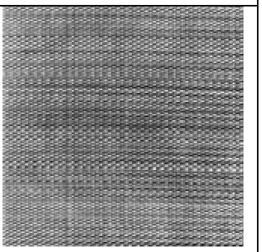
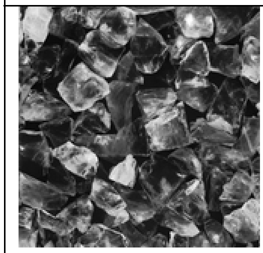
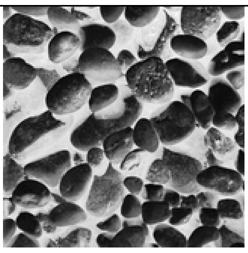
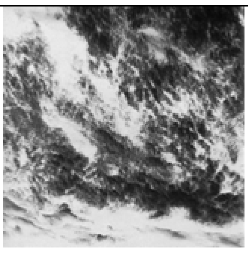
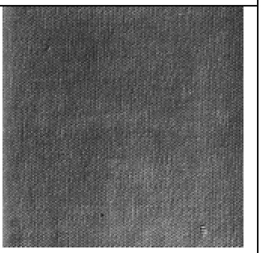
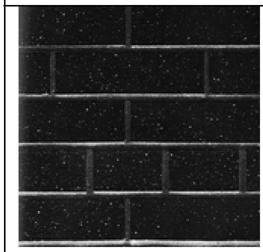
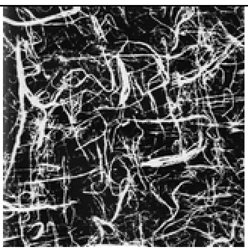
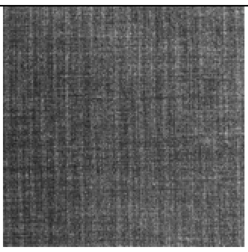
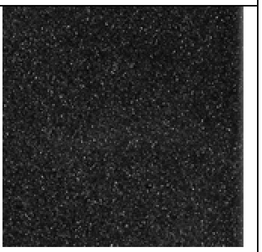
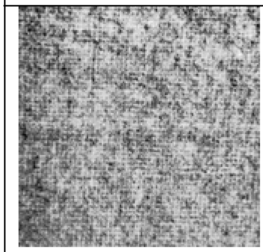
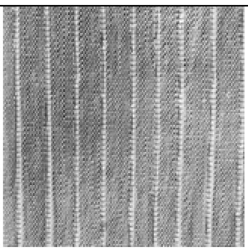
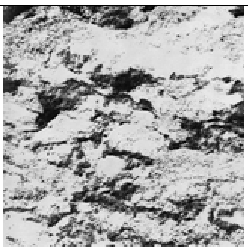
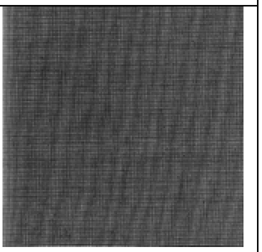
			
0,002181	0,0028494	0,012652	0,018372
			
0,019923	0,021419	0,024523	0,029034
			
0,030041	0,033109	0,035272	0,038315
			
0,043165	0,043599	0,043816	0,058781
			
0,074304	0,07729	0,089946	0,090257

Table 6-2 (continued)

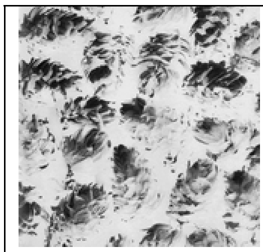
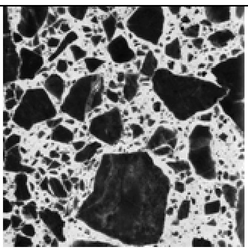
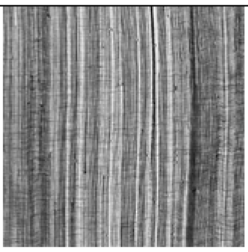
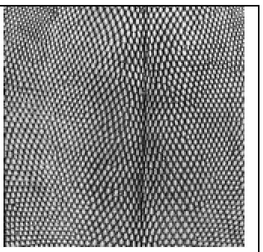
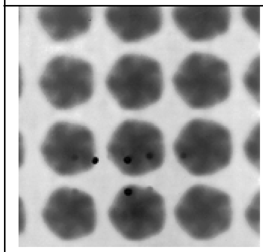
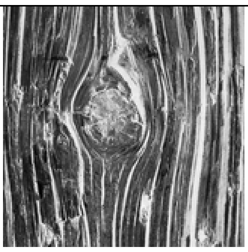
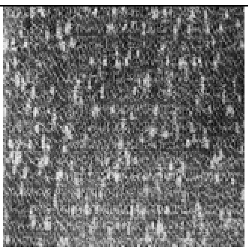
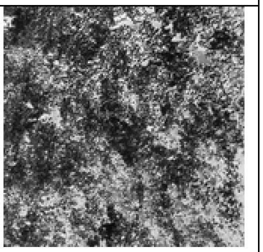
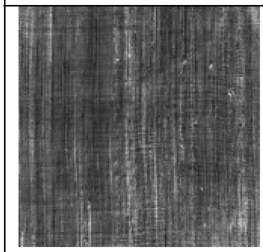
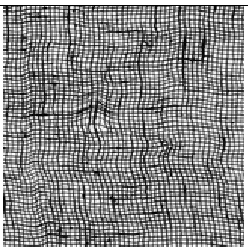
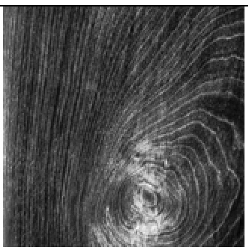
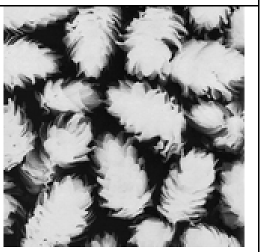
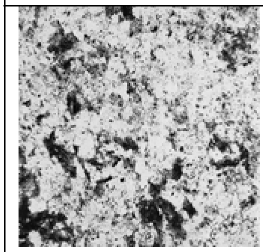
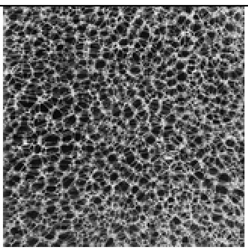
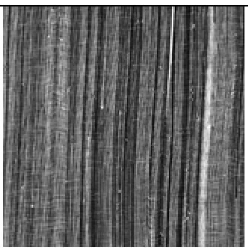
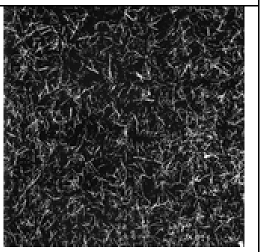
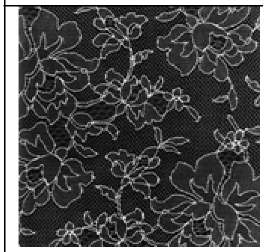
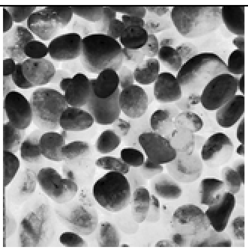
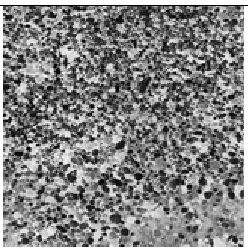
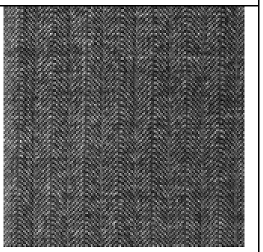
			
0,090403	0,09415	0,10572	0,1103
			
0,11418	0,12586	0,12599	0,13374
			
0,14324	0,15241	0,15311	0,15893
			
0,15965	0,16745	0,17718	0,18152
			
0,19683	0,19953	0,21049	0,21498

Table 6-2 (continued)

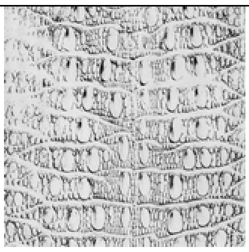
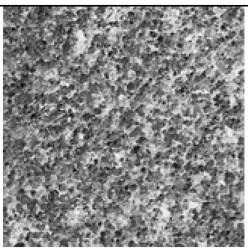
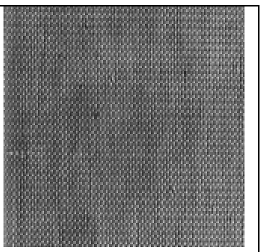
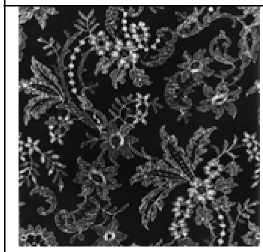
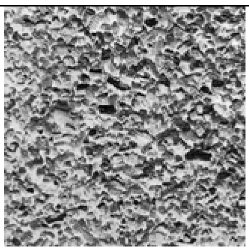
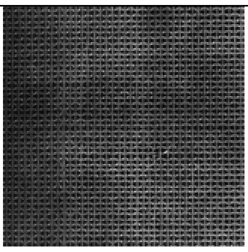
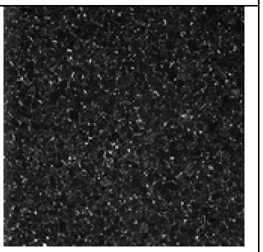
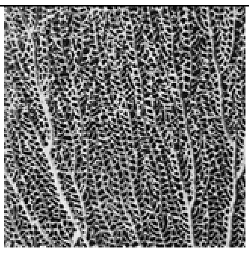
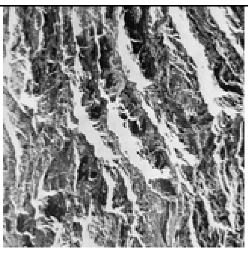
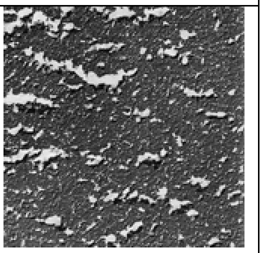
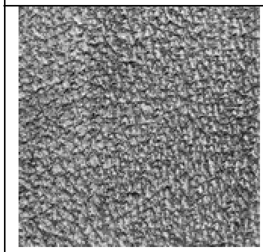
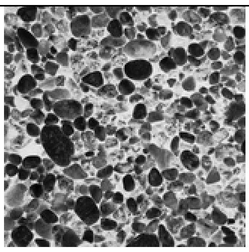
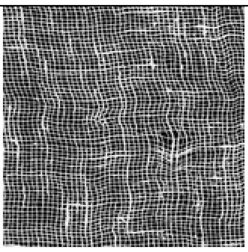
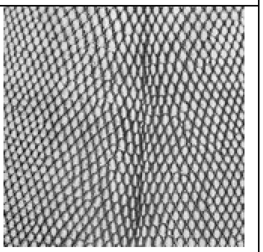
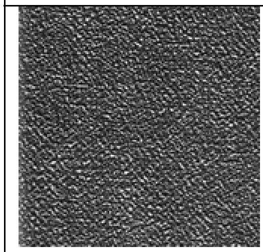
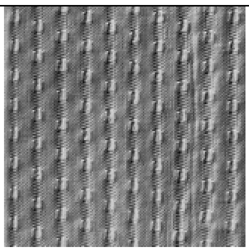
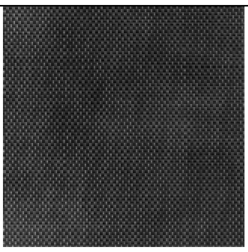
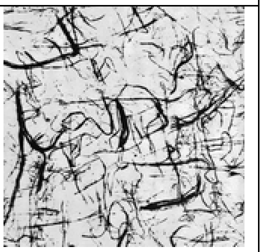
			
0,21612	0,2219	0,23084	0,2323
			
0,23728	0,25387	0,26957	0,27164
			
0,28344	0,30761	0,31476	0,34097
			
0,3713	0,38033	0,38151	0,46421
			
0,47999	0,62065	0,64512	0,6896

Table 6-2 (continued)

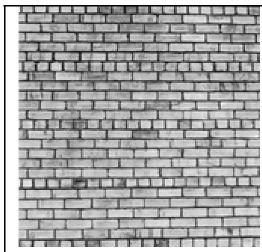
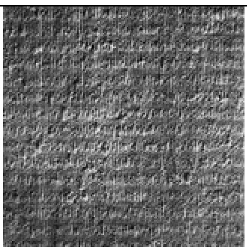
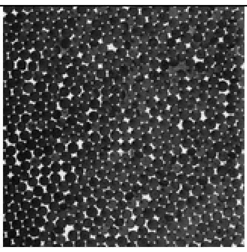
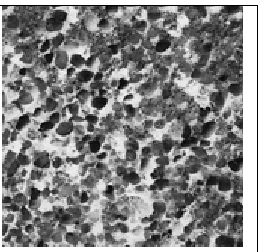
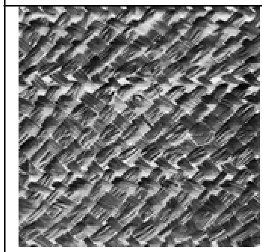
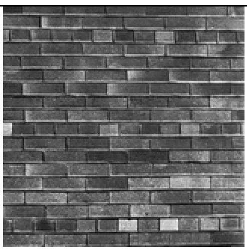
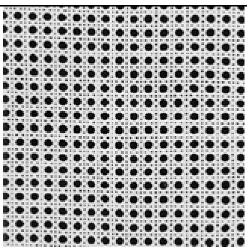
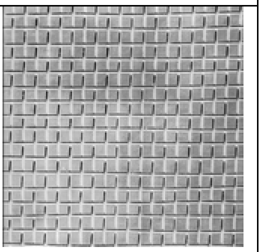
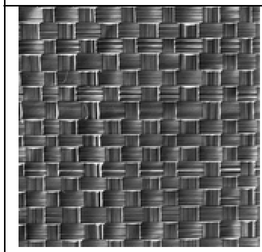
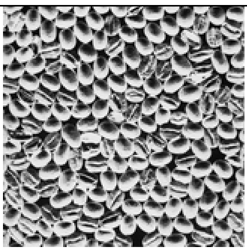
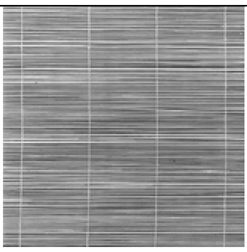
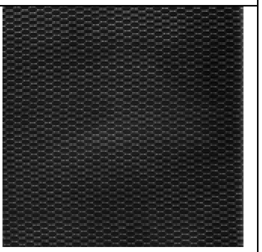
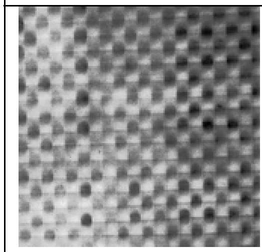
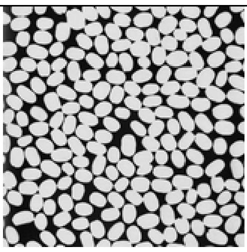
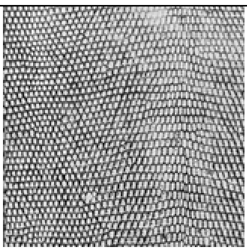
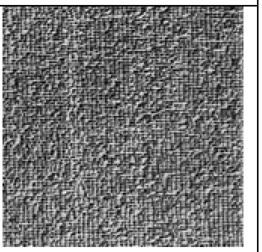
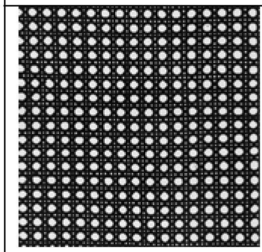
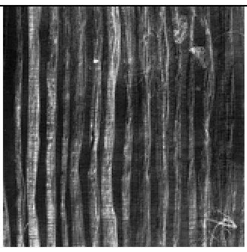
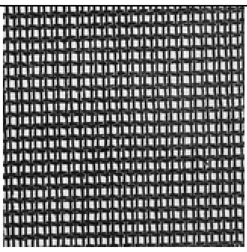
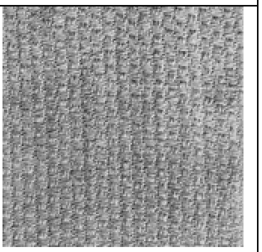
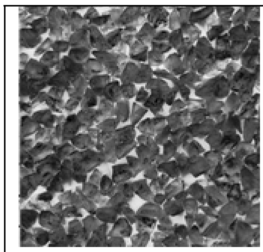
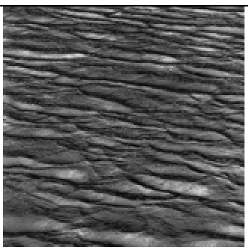
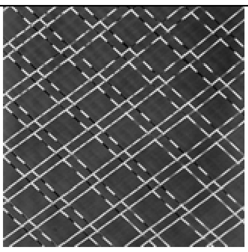
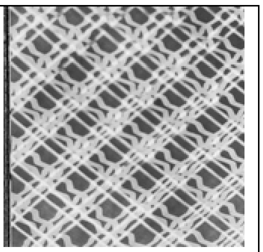
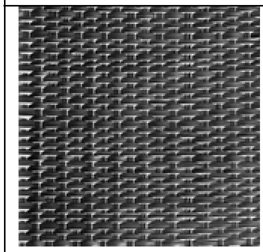
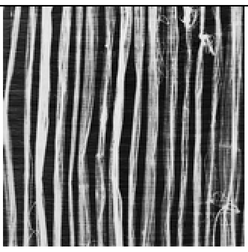
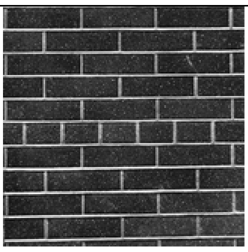
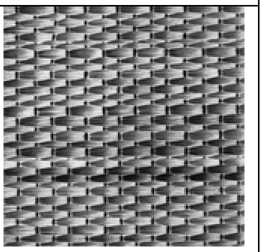
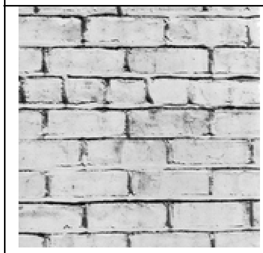
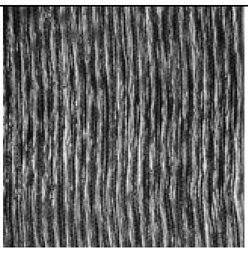
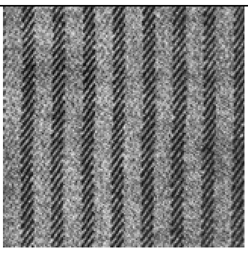
			
0,70345	0,72017	0,73409	0,73795
			
0,75637	0,75672	0,75753	0,77216
			
0,78162	0,79331	0,79443	0,81717
			
0,8177	0,81809	0,83796	0,83822
			
0,8508	0,86825	0,8829	0,88886

Table 6-2 (continued)

			
0,89037	0,89179	0,89264	0,90086
			
0,91836	0,92099	0,92449	0,92785
			
0,9413	0,9618	0,99174	

Evaluating the results given in Table-6.1 and Table-6.2, it is obvious that, talking with obtained results, our proposed measure successfully differentiates textures according to its regularities similar to human quality perception. While qualitative evaluation, it is also perceived that some of the results are wrong without controversial. However, the true classification rate can be evaluated as having success more than 90%, the reasons of these faults have to be declared. For determining reliable texture regularity measure the calculated contrast function has to catch more than two periods of the pattern. In case of missing periods, resulting measure is not reliable. For example, on last row, 2nd column of Table-6.1 and 9th row, 1st column of Table-6.2 has occurrences like this. Examining the results, it can be noticed that some measure scores for regular textures is less than expected. This drop in regularity score is because of median filtering the contrast function. In

Figure-6.5, an example of this situation is presented and it can be seen clearly how the expected result can be affected.

During classification in deterministic method, two of six features are determined from texture regularity, but these are not enough alone for true classification. We could not yet find opportunity for observing how much fine tuning on texture regularity affect classification results; because of required calculation times are too long. However we do expect that whereas this measure produces relatively similar results, fine tuning has minor affect on classification.

6.4 Dynamic Texture Classification

In this section, the results of comparison between two dynamic texture classification approaches are presented, where stochastic approach is explained in chapter 4 and deterministic approach is explained in chapter 5 of this thesis.

The database, constructed as explained in section 6.2, has 384 dynamic textures, each consists of 60 frames of 48x48 images. In other words, we have video sequences at size 48x48x60. We call each of these as a set. These sets are grouped in 32 classes and 13 categories as given in Table-6.3.

The sets are divided into two groups and named as : test sets and training sets each having 192 sets. The classifications are realized on these sets.

Stochastic and deterministic methods are different from each other while classifying sets. Stochastic method compares sets with sets, while deterministic method compares sets with classes. This distinction on classification methodology produces an advantage for deterministic method.

Table 6-3: Sets, Classes and Categories in Database.

Categories	Classes	Sets in Class	Sets in Category
Boiling water	boil-heavy	12	80
	boil-heavy 2	12	
	boil-light	12	
	boil-light 2	12	
	boil-light 3	12	
	boil-side	8	
	boil-side 2	12	
Escalator	escalator	12	12
Fire	fire	20	36
	fire 2	16	
Flags	flags	16	24
	flags 2	8	
Fountain	fountain	8	8
Laundry	laundry	12	12
Plastic	plastic	12	36
	plastic 2	12	
	plastic 3	12	
River	river	16	56
	river 2	16	
	river-far	12	
	river-far 2	12	
Shower	shower	8	8
Haze	smoke	12	20
	smoke 2	8	
	steam	12	32
	steam 2	8	
	steam 3	12	
Stripes	stripes	12	20
	stripes 2	8	
Toilet	toilet	12	12
Trees	trees	16	28
	trees 2	12	

Table-6.4 and Table-6.5 shows the classification results of methods on boiling water category. Results of classifications are presented as confusion matrices where rows are showing test sets and columns are showing how many times the test set is classified in training sets as its closest neighbor.

Table 6-4: Confusion matrix of deterministic method on boiling water category.

	boil-heavy	boil-heavy 2	boil-light	boil-light 2	boil-light 3	boil-side	boil-side 2
boil-heavy	4	2	0	0	0	0	0
boil-heavy 2	0	6	0	0	0	0	0
boil-light	0	0	6	0	0	0	0
boil-light 2	0	0	2	4	0	0	0
boil-light 3	0	0	0	0	6	0	0
boil-side	2	0	0	0	0	2	0
boil-side 2	0	0	0	0	0	3	3

Table 6-5: Confusion matrix of stochastic method on boiling water category.

	boil-heavy	boil-heavy 2	boil-light	boil-light 2	boil-light 3	boil-side	boil-side 2
boil-heavy	2	2	0	0	0	0	2
boil-heavy 2	2	3	0	0	0	1	0
boil-light	0	0	1	3	2	0	0
boil-light 2	0	0	2	3	1	0	0
boil-light 3	0	0	1	2	3	0	0
boil-side	0	2	0	0	0	2	0
boil-side 2	1	2	0	0	1	0	2

When we examined the results in Table-6.4 and Table-6.5, it is clear that deterministic method produces better results. Notice that deterministic method classify sets by looking at distances between the features of the test set and the mean of the features of the sets in classes, while stochastic method classify sets by looking at distances between models of sets. However a classification within a category is not required in most of applications, it gives us information about behaviors of classification methods.

Table 6-6: Confusion matrix of deterministic method on boiling water and river categories.

	boil-heavy	boil-heavy 2	boil-light	boil-light 2	boil-light 3	boil-side	boil-side 2	river	river 2	river-far	river-far 2
boil-heavy	4	2	0	0	0	0	0	0	0	0	0
boil-heavy 2	0	6	0	0	0	0	0	0	0	0	0
boil-light	0	0	6	0	0	0	0	0	0	0	0
boil-light 2	0	0	2	4	0	0	0	0	0	0	0
boil-light 3	0	0	0	0	6	0	0	0	0	0	0
boil-side	2	0	0	0	0	2	0	0	0	0	0
boil-side 2	0	0	0	0	0	3	3	0	0	0	0
river	0	0	0	0	0	0	0	8	0	0	0
river 2	0	0	0	0	0	0	0	1	4	0	3
river-far	0	0	0	0	0	0	0	0	0	6	0
river-far 2	1	0	0	0	0	0	0	0	3	0	2

Table 6-7: Confusion matrix of stochastic method on boiling water and river categories.

	boil-heavy	boil-heavy 2	boil-light	boil-light 2	boil-light 3	boil-side	boil-side 2	river	river 2	river-far	river-far 2
boil-heavy	2	2	0	0	0	0	2	0	0	0	0
boil-heavy 2	2	3	0	0	0	1	0	0	0	0	0
boil-light	0	0	1	3	2	0	0	0	0	0	0
boil-light 2	0	0	2	3	1	0	0	0	0	0	0
boil-light 3	0	0	1	2	3	0	0	0	0	0	0
boil-side	0	2	0	0	0	2	0	0	0	0	0
boil-side 2	1	2	0	0	1	0	2	0	0	0	0
river	0	0	0	0	0	0	0	2	6	0	0
river 2	0	0	0	0	0	0	0	3	5	0	0
river-far	0	0	0	0	0	0	0	0	3	3	0
river-far 2	0	0	0	0	0	0	0	0	2	3	1

Table-6.6 and Table-6.7 shows the classification results when the sets are extended with river category. We have seen previously that deterministic method was successful than the stochastic method on classifying sets between classes within a category. Extending our tests on two categories shows us that stochastic method is now successful than deterministic method on classifying sets between categories. As it is observed in Table-6.6 a river-far 2 sequence is named once as boil-heavy sequence with deterministic method, whereas there is no miss out of category with stochastic method as it can be seen in Table-6.7.

Table 6-8: Confusion matrix of deterministic method on some mixed classes.

	boil-heavy	boil-light	escalator	fire	flags	laundry	plastic	river	toilet	trees
boil-heavy	2	0	0	1	0	0	0	0	3	0
boil-light	0	6	0	0	0	0	0	0	0	0
escalator	0	0	6	0	0	0	0	0	0	0
fire	0	0	0	7	2	0	0	0	0	1
flags	0	1	0	0	3	4	0	0	0	0
laundry	0	0	0	0	0	6	0	0	0	0
plastic	0	0	0	0	0	0	5	0	0	1
river	0	0	0	0	0	0	0	8	0	0
toilet	0	0	0	1	0	0	0	0	5	0
trees	0	1	0	0	0	0	2	0	1	4

Table 6-9: Confusion matrix of stochastic method on some mixed classes.

	boil-heavy	boil-light	escalator	fire	flags	laundry	plastic	river	trees
boil-heavy	4	0	0	0	0	2	0	0	0
boil-light	0	6	0	0	0	0	0	0	0
escalator	0	0	6	0	0	0	0	0	0
fire	0	0	0	10	0	0	0	0	0
flags	2	0	0	0	3	3	0	0	0
laundry	1	0	0	0	0	5	0	0	0
plastic	0	0	0	0	0	0	3	0	3
river	0	0	0	0	0	0	0	8	0
toilet	0	0	0	0	6	0	0	0	0
trees	0	0	0	0	0	0	1	0	7

What happen when we use a mix set of classes each showing different characteristics from each other? In Table-6.8 and Table-6.9 the results of a test like this are shown. At first glance it attracts an attention on that boil-heavy and boil-light sequences are in same category, but they show different characteristics that we can also categorize them in a different category.

Examine Table-6.4 and Table-6.5 and see that two methods can successfully differentiate these sub-categories from each other. In Table-6.8 and Table-6.9 we start to see that both methods play different behaviors depending on characteristics of sets. Both methods have reached to a %100 percent true classification in 4 different classes. When we examine the results deeply, we can say that stochastic method succeeds more than deterministic method on case of a test which has sample sets from different categories, except a %100 of miss-classification on toilet sequence. Notice that, the reason of this miss-classification is that the characteristic of toilet sequence is very similar to flags sequence in terms of models which the stochastic method calculates distances using these models.

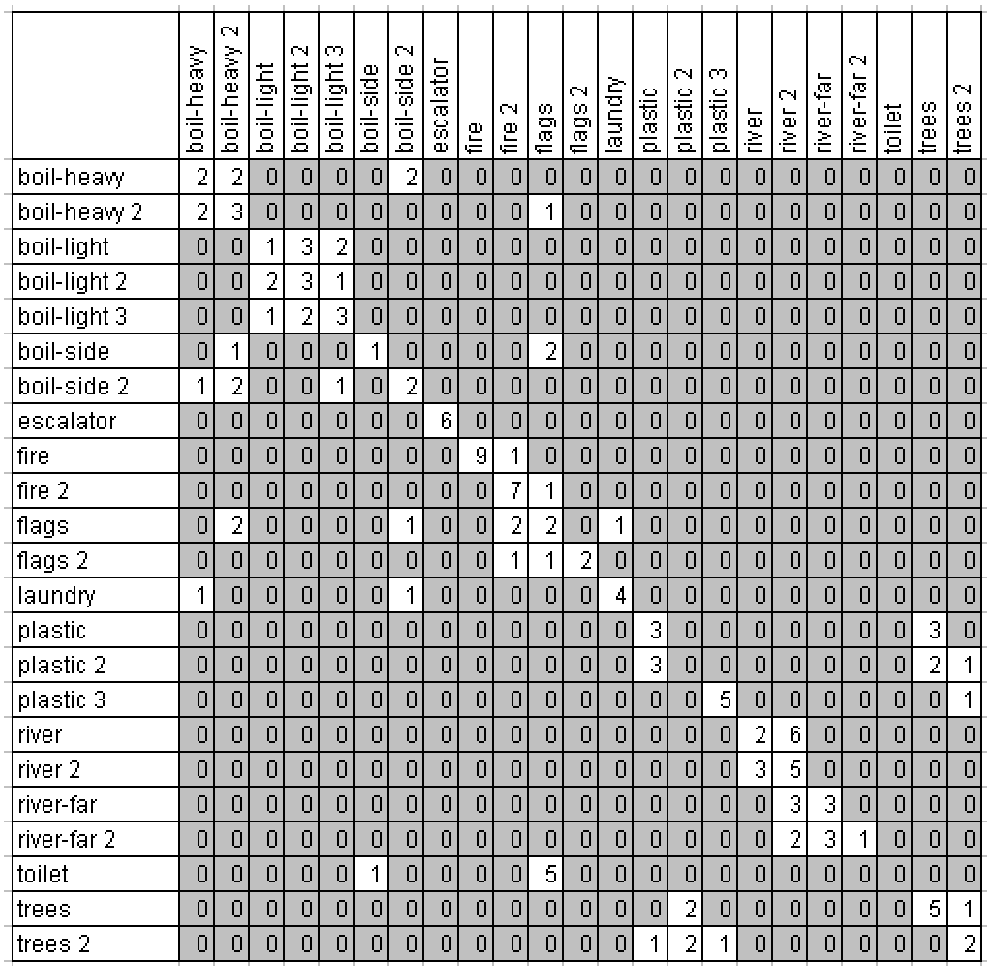
The results of mixed sets of classes, when we add other sets of classes of the categories used previously, are shown in Table-6.10 and Table-6.11. In escalator sequence, we see that both methods have classified it with no misses. This is because of that the escalator has totally different characteristics than the other categories, or in other words, it can be said that the features and models of methods can fully represent the characteristic of this dynamic texture. In both results we also see that both methods have truly classified most of the categories, where they have also missed some of the sets. For example, deterministic method is having problems while classifying sets from flags and laundry categories and stochastic method is having problems while classifying sets from flags, toilet, plastic and trees categories. In case of miss-classifying the plastic and trees categories with each other, it can be comprehensible for stochastic method that these categories show very similar characteristics; both of them are stationary processes.

Table 6-10: Confusion matrix of deterministic method on some mixed classes.

	boil-heavy	boil-heavy 2	boil-light	boil-light 2	boil-light 3	boil-side	boil-side 2	escalator	fire	fire 2	flags	flags 2	laundry	plastic	plastic 2	plastic 3	river	river 2	river-far	river-far 2	toilet	trees	trees 2
boil-heavy	1	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2	0	0
boil-heavy 2	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light 2	0	0	2	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light 3	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-side	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
boil-side 2	0	0	0	0	0	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
escalator	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
fire	0	0	0	0	0	0	0	0	6	0	2	0	0	0	0	0	0	0	0	0	0	0	2
fire 2	0	0	0	0	0	0	0	0	0	7	0	0	1	0	0	0	0	0	0	0	0	0	0
flags	0	0	0	1	0	0	0	0	0	0	3	1	1	0	0	0	0	2	0	0	0	0	0
flags 2	0	0	0	0	0	0	0	0	0	3	1	0	0	0	0	0	0	0	0	0	0	0	0
laundry	0	0	0	0	0	0	0	0	0	1	0	0	3	0	0	0	0	0	0	2	0	0	0
plastic	0	0	0	0	0	0	0	0	0	0	0	0	0	4	1	0	0	0	0	0	0	1	0
plastic 2	0	0	0	0	1	0	0	0	1	0	0	0	0	0	4	0	0	0	0	0	0	0	0
plastic 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0
river	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0
river 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	0	3	0	0	0
river-far	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0
river-far 2	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	1	0	2	0	0	0
toilet	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	4	0	0
trees	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	6
trees 2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	4

Finally, in Table-6.12 and Table-6.13 the results of the classification of all sets in database are shown. In these results, other than we have stated before, we can easily observe that both methods have problems on specific sets. Deterministic method is having a lot of problems with sets from flags and stripes categories, while it has also some problems with sets from fountain, laundry and trees categories. Stochastic method is having a lot of problems with sets from flags, fountain, shower, stripes and toilet categories, while it has also some problems with plastic and trees categories. Both methods are successful on classifying the sets from remaining categories, while they have also miss-classifications in terms of classifying as classes.

Table 6-11: Confusion matrix of stochastic method on some mixed classes.



	boil-heavy	boil-heavy 2	boil-light	boil-light 2	boil-light 3	boil-side	boil-side 2	escalator	fire	fire 2	flags	flags 2	laundry	plastic	plastic 2	plastic 3	river	river 2	river-far	river-far 2	toilet	trees	trees 2
boil-heavy	2	2	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-heavy 2	2	3	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
boil-light	0	0	1	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light 2	0	0	2	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light 3	0	0	1	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-side	0	1	0	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
boil-side 2	1	2	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
escalator	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
fire	0	0	0	0	0	0	0	0	9	1	0	0	0	0	0	0	0	0	0	0	0	0	0
fire 2	0	0	0	0	0	0	0	0	0	7	1	0	0	0	0	0	0	0	0	0	0	0	0
flags	0	2	0	0	0	0	1	0	0	2	2	0	1	0	0	0	0	0	0	0	0	0	0
flags 2	0	0	0	0	0	0	0	0	0	1	1	2	0	0	0	0	0	0	0	0	0	0	0
laundry	1	0	0	0	0	0	1	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0
plastic	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	3	0
plastic 2	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	2	1
plastic 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	1
river	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	6	0	0	0	0	0
river 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	5	0	0	0	0	0
river-far	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	0	0	0	0
river-far 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3	1	0	0	0
toilet	0	0	0	0	0	1	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0
trees	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	5	1
trees 2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	1	0	0	0	0	0	0	2

When we have to evaluate methods on classifying all sets in database, it can be said that the deterministic method is more successful than the stochastic method. The results also show that both methods have a good discrimination power on dynamic textures. However the classification results give a lot of detailed information about these methods classification successes, their success must also be calculated as true recognition rates.

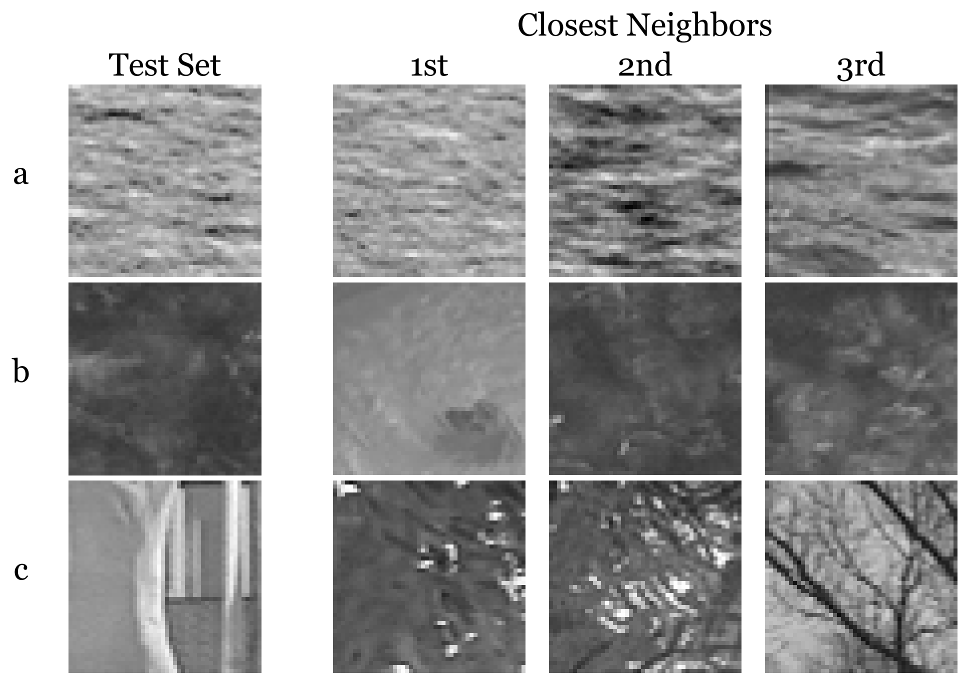


Figure 6-8: *a) A true classification, b) A semi-true classification, c) A miss-classification, using deterministic method on sets shown in Table-6.13.*

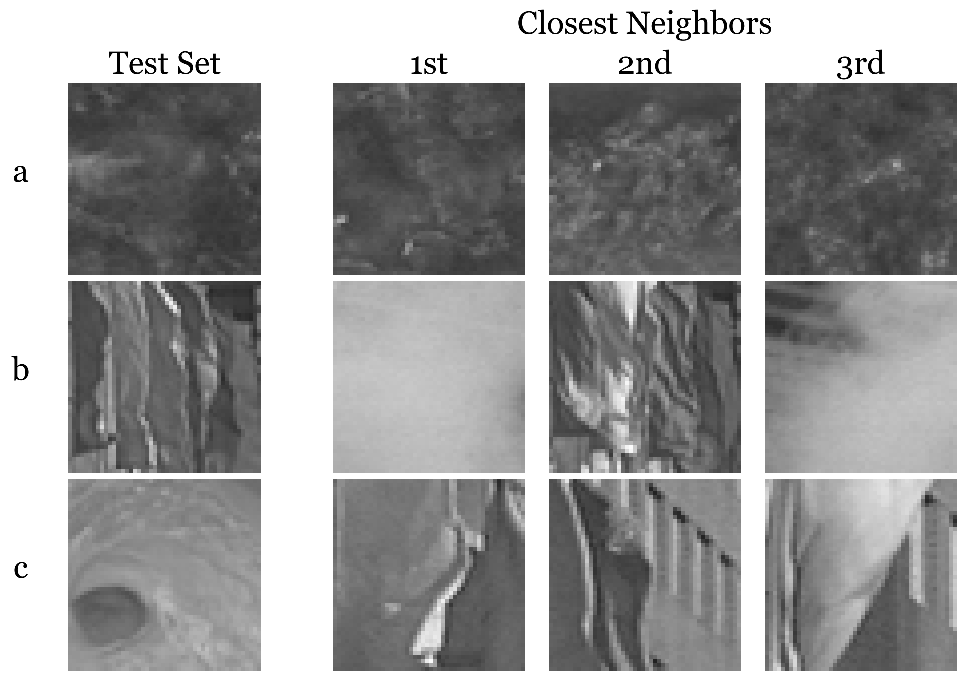


Figure 6-9: *a) A true classification, b) A semi-true classification, c) A miss-classification, using stochastic method on sets shown in Table-6.14.*

Table 6-12: Confusion matrix of deterministic method on all classes.

	boil-heavy	boil-heavy 2	boil-light	boil-light 2	boil-light 3	boil-side	boil-side 2	escalator	fire	fire 2	flags	flags 2	fountain	laundry	plastic	plastic 2	plastic 3	river	river 2	river-far	river-far 2	shower	smoke	smoke 2	steam	steam 2	steam 3	stripes	stripes 2	toilet	trees	trees 2
boil-heavy	1	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0
boil-heavy 2	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light 2	0	0	2	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light 3	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-side	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
boil-side 2	0	0	0	0	0	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
escalator	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
fire	0	0	0	0	0	0	0	0	6	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2
fire 2	0	0	0	0	0	0	0	0	0	7	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
flags	0	0	0	1	0	0	0	0	0	0	3	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	1	0	0	0
flags 2	0	0	0	0	0	0	0	0	0	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
fountain	0	0	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
laundry	0	0	0	0	0	0	0	0	0	1	0	0	0	3	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
plastic	0	0	0	0	0	0	0	0	0	0	0	0	1	0	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
plastic 2	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
plastic 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
river	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
river 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	0	3	0	0	0	0	0	0	0	0	0	0	0
river-far	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0
river-far 2	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	1	0	0	0	2	0	0	0	0	0	0	0
shower	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	1	0	0
smoke	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	3	1	0	0	0	0	0
smoke 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	1	0	0	0	0	0	0
steam	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0
steam 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2	0	0	0	0	0	0
steam 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	4	0	0	0	0	0
stripes	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1	0	0
stripes 2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2	0	0	0
toilet	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0
trees	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3
trees 2	0	0	0	0	0	0	0	0	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2

For calculating the recognition rates, firstly, the terms of true classification has to be defined. We have defined four kinds of true classification as:

1. If the class of the 1st closest neighbor of the classified sets is same as the class of the test set,
2. If the category of the 1st closest neighbor of the classified sets is same as the category of the test set,

3. If the class of one of the 2 closest neighbors of the classified sets is same as the class of the test set,
4. If the category of one of the 2 closest neighbors of the classified sets is same as the category of the test set.

The recognition rates are measured, as described above, for the given sets in tables and shown in following tables. The true classification definitions are named as rating method in tables.

By defining four different rating methods we aimed to clarify the capabilities of classification methods. It will be meaningful to watch the score of 2nd rating method, which calculates rating according to category. Think of a real application that you have to distinguish river from road or fire from smoke. It is obvious that the application needs classifying of textures according to categories. 1st rating is calculated for catching sensitivity of the methods. 3rd and 4th ratings are calculated for interrogating if the method has ability to truly classify the missed sets on its 2nd closest neighbors. In a real application one can search for two occurrences in closest neighbors and create a decision of taking the first two times occurred sets in closest neighbors as the result of classification. As we emphasize the 2nd result as the main score, it is shaded in tables except in Table-6.14. In Table-6.14, there is only one category and in that classification it is aimed to expose the sensitivity of the method within category classification.

Table 6-13: Confusion matrix of stochastic method on all classes.

	boil-heavy	boil-heavy 2	boil-light	boil-light 2	boil-light 3	boil-side	boil-side 2	escalator	fire	fire 2	flags	flags 2	fountain	laundry	plastic	plastic 2	plastic 3	river	river 2	river-far	river-far 2	shower	smoke	smoke 2	steam	steam 2	steam 3	stripes	stripes 2	toilet	trees	trees 2
boil-heavy	2	2	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-heavy 2	2	3	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light	0	0	1	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light 2	0	0	2	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-light 3	0	0	1	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-side	0	1	0	0	0	1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
boil-side 2	1	2	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
escalator	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
fire	0	0	0	0	0	0	0	0	8	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
fire 2	0	0	0	0	0	0	0	0	0	6	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
flags	0	2	0	0	0	0	1	0	0	2	2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
flags 2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	0	0
fountain	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0
laundry	1	0	0	0	0	0	1	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
plastic	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	2	0
plastic 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1
plastic 3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
river	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	5	0	0	0	0	1	0	0	0	0	0	0	0	0
river 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	5	0	0	0	1	1	0	0	0	0	0	0	0	0
river-far	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	1	0	0	0	2	0	0	0	0	0	0	0	0
river-far 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	1	0	0	1	0	0	0	0	0	0	0	0
shower	0	0	0	0	0	0	0	0	1	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
smoke	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	2	0	0	0	0	0	3	0	0	0
smoke 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	2	0	0	0	0	0	0	0	0	0
steam	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2	0	1	0	0	0	0	0
steam 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	1	0	0	0
steam 3	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	3	0	1	0	0	0	0	0
stripes	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	0	1	0	0	0	0
stripes 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0
toilet	0	0	0	0	0	1	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
trees	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	1
trees 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2

Table 6-14: Recognition rates of both methods under test sets given in Table-6.4 and Table-6.5.

Rating Method	Recognition Rates	
	Stochastic Method	Deterministic Method
1	% 40	% 78
2	% 100	% 100
3	% 60	% 95
4	% 100	% 100

Table 6-15: Recognition rates of both methods under test sets given in Table-6.6 and Table-6.7.

Rating Method	Recognition Rates	
	Stochastic Method	Deterministic Method
1	% 40	% 75
2	% 100	% 99
3	% 65	% 94
4	% 100	% 99

Table 6-16: Recognition rates of both methods under test sets given in Table-6.8 and Table-6.9.

Rating Method	Recognition Rates	
	Stochastic Method	Deterministic Method
1	% 74	% 74
2	% 74	% 74
3	% 84	% 83
4	% 84	% 83

Table 6-17: Recognition rates of both methods under test sets given in Table-6.10 and Table-6.11.

Rating Method	Recognition Rates	
	Stochastic Method	Deterministic Method
1	% 48	% 64
2	% 78	% 78
3	% 64	% 79
4	% 86	% 90

Table 6-18: Recognition rates of both methods under test sets given in Table-6.12 and Table-6.13.

Rating Method	Recognition Rates	
	Stochastic Method	Deterministic Method
1	% 37	% 60
2	% 65	% 75
3	% 53	% 77
4	% 78	% 89

6.5 Computational Complexity

A Pentium-IV, 3.0 GHz CPU having 512 MB memory which are placed on a motherboard with a state of the art dual-RAM technology computer is used in simulating our studies. As we have stated before, the database consists of 384 dynamic textures. A first step before classification, the model parameters of dynamic textures in stochastic method and features of dynamic textures in deterministic method must be calculated. Calculating model parameters for all dynamic textures in database take approximately 8 minutes, while calculating features for all dynamic textures in database take approximately 40 hours. Once the model parameters and features are calculated the classifications finish in seconds, where classification with deterministic method is faster because of its simplicity.

6.6 Dynamic Texture Synthesis

However synthesizing dynamic textures is out of scope of this thesis, the power of the stochastic method described in chapter 4 of this thesis must be shown. Remember that a dynamic texture can be represented, in other words modeled, with only three parameters A, B and C as given with equations (4.1) and (4.9). Using these model parameters it is possible to synthesize "infinite length" texture sequences from a typically "short" input sequence by just drawing IID samples $v(t)$ from a Gaussian distribution. Some of the results are shown in Figure-6.10, Figure-6.11, Figure-6.12, Figure-6.13, Figure-6.14 and Figure-6.15.

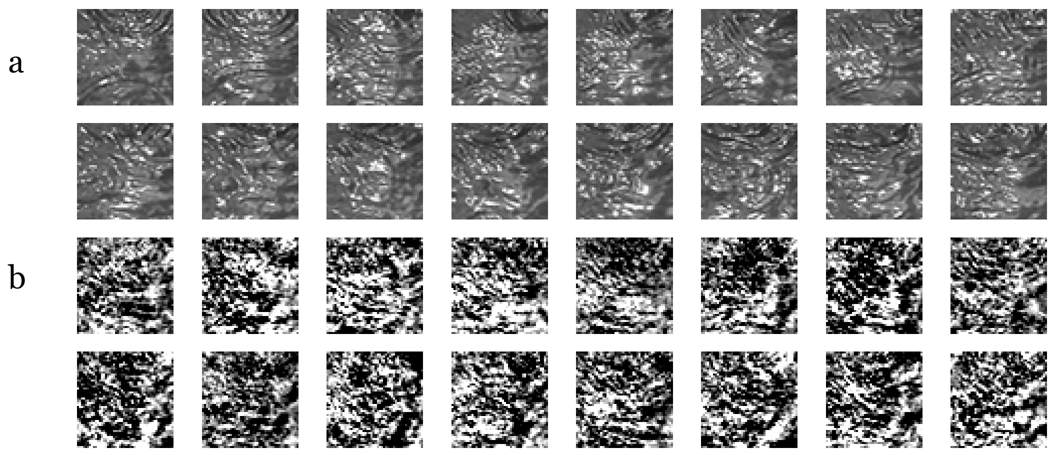


***Figure 6-10:** a) 16 frames of light boiling water sequence and b) synthesized frames.*

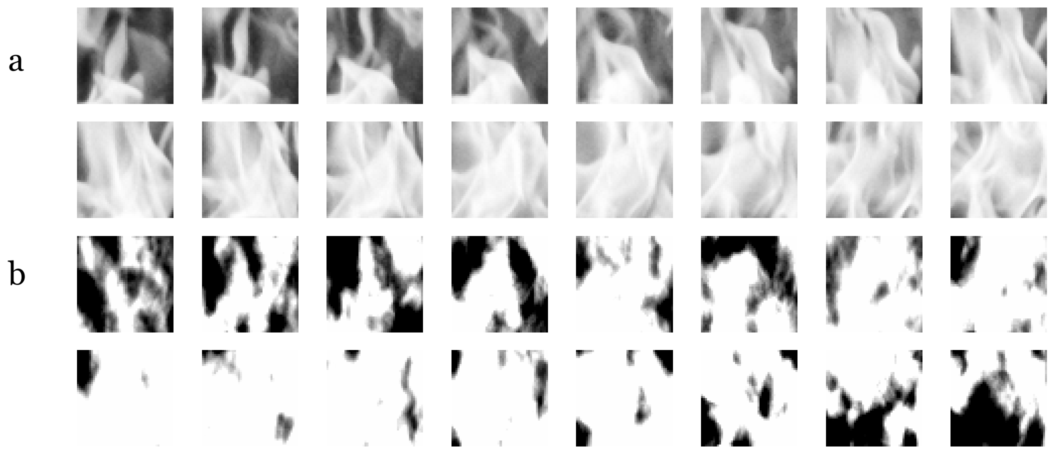


Figure 6-11: *a) 16 frames of fire sequence and b) synthesized frames.*

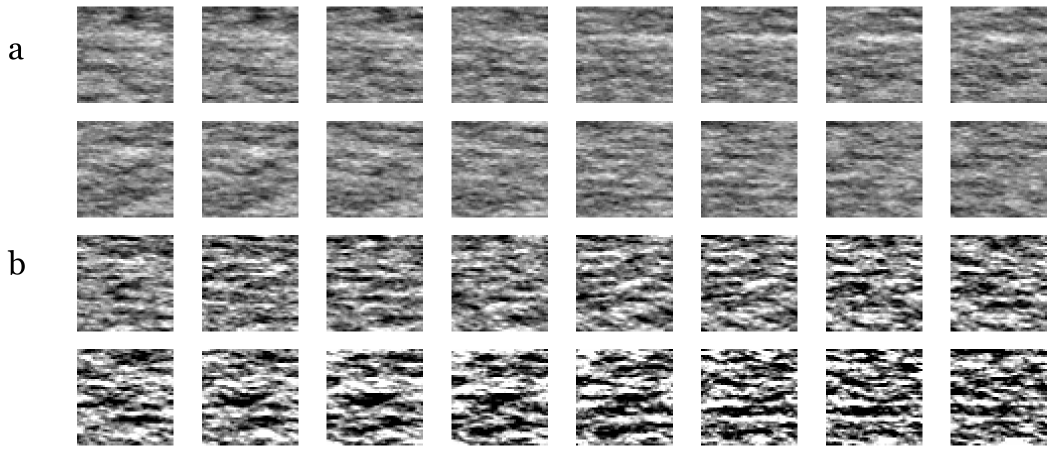


Figure 6-12: *a) 16 frames of river sequence and b) synthesized frames.*

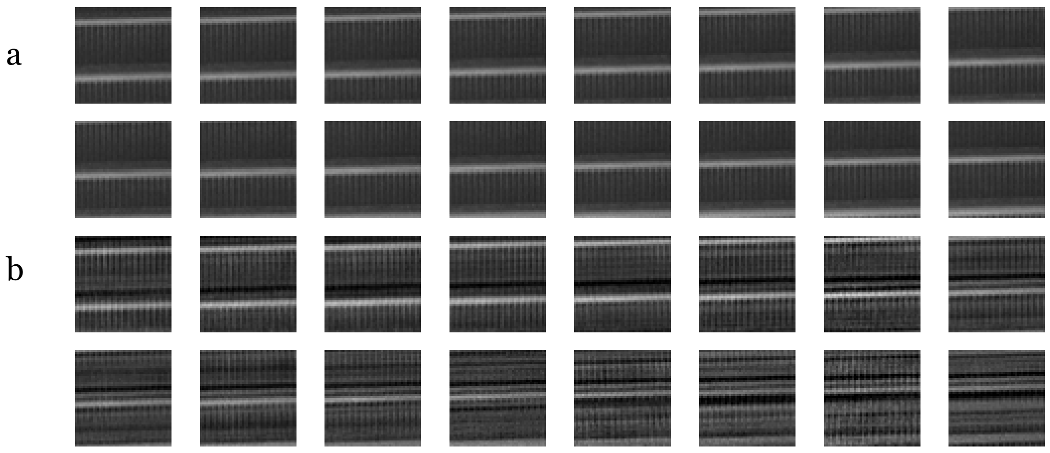


Figure 6-13: *a) 16 frames of escalator sequence and b) synthesized frames.*

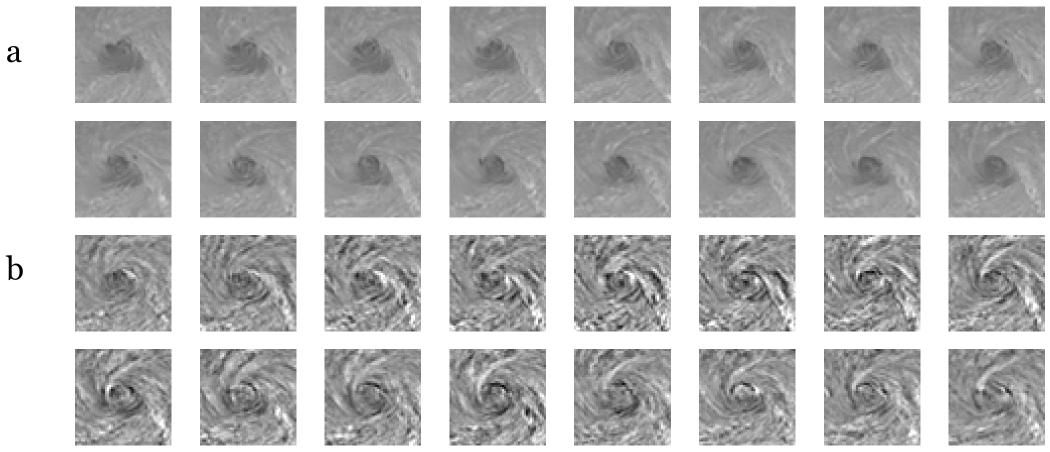


Figure 6-14: *a) 16 frames of toilet sequence and b) synthesized frames.*

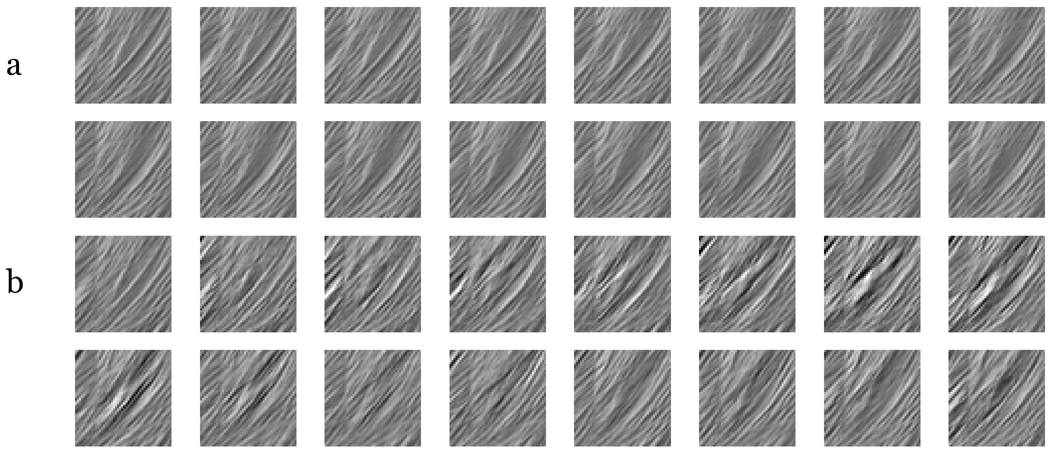
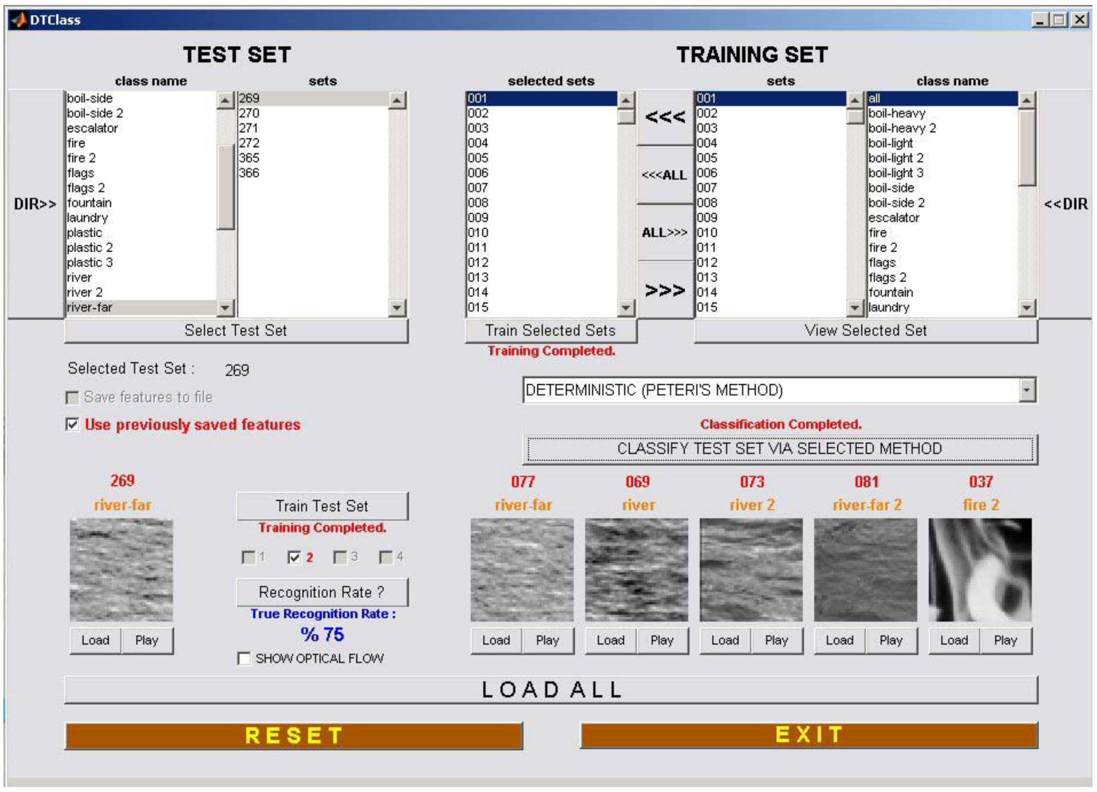


Figure 6-15: *a) 16 frames of plastic sequence and b) synthesized frames.*



Training Completed.

Selected Test Set : 269

Save features to file

[✓ Use previously saved features]

DETERMINISTIC (PETERI'S METHOD)

Classification Completed.

269 river-far

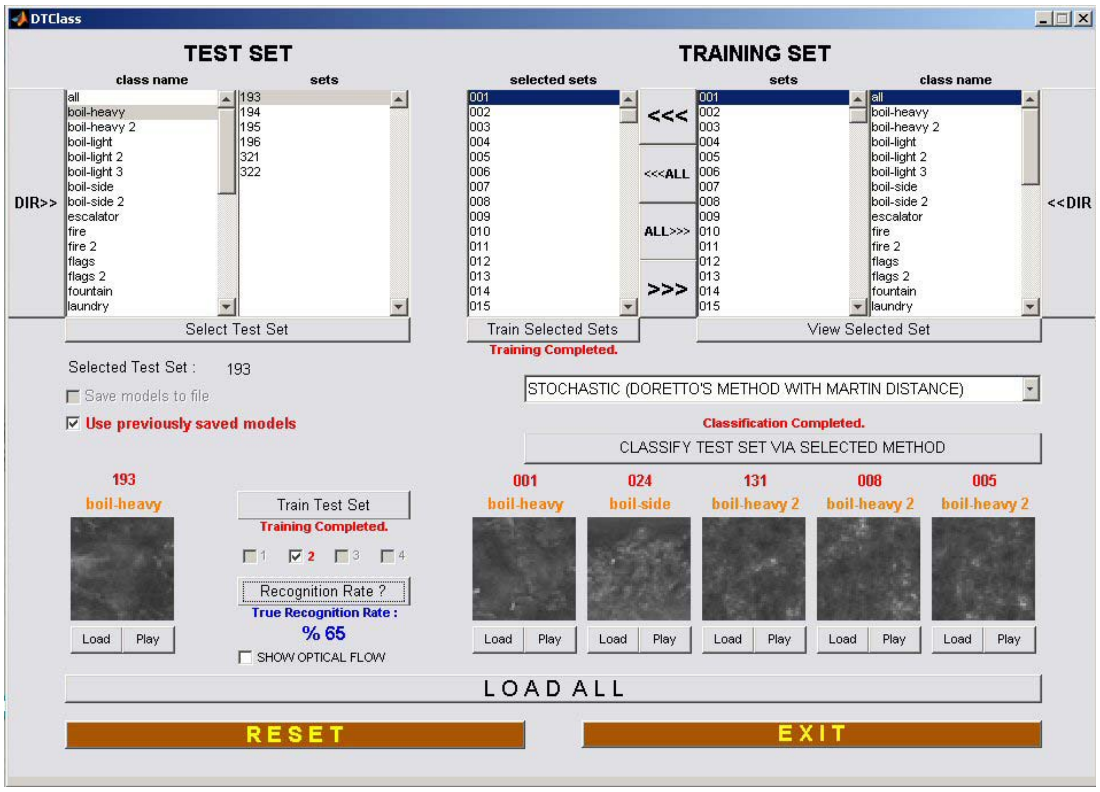
Training Completed.

1 [✓ 2] 3 4

True Recognition Rate : % 75

077 river-far | 069 river | 073 river 2 | 081 river-far 2 | 037 fire 2

Figure 6-16: *A view of our dynamic texture classification program when the results presented in Figure-6.8 (a) and in Table-6.18 are calculated.*



Selected Test Set : 193

☑ Use previously saved models

Training Completed.

STOCHASTIC (DORETTO'S METHOD WITH MARTIN DISTANCE)

Classification Completed.

193 boil-heavy

Training Completed.

True Recognition Rate : % 65

001 boil-heavy

024 boil-side

131 boil-heavy 2

008 boil-heavy 2

005 boil-heavy 2

***Figure 6-17:** A view of our dynamic texture classification program when the results presented in Figure-6.9 (a) and in Table-6.18 are calculated.*

CHAPTER 7

SUMMARY & CONCLUSIONS

In this thesis, the dynamic textures are studied in detail. Our aim is to analyse and characterize these dynamic textures using several techniques. For this purpose, two successful methods in literature are selected. One approaches the subject using stochastic techniques and the other approaches using deterministic techniques.

In stochastic approach, modeling dynamic textures is defined as a system identification problem. Dynamic textures are represented as ARMA models and the model parameters are calculated as a result of learning phase. It is shown that dynamic textures can be represented with only three model parameters. It is also shown that using these model parameters it is possible to synthesize “infinite length” texture sequences from a typically “short” input sequence by just drawing IID samples from a gaussian distribution. However, it is not aimed in this thesis, some results of synthesing dynamic textures are shown. The classification of dynamic textures using model parameters is explained and realized.

In deterministic approach, the spatial and temporal features of dynamic textures are calculated using normal flow and texture regularity. It can be said that it is possible to extract accurate features that can be used for classification of dynamic textures.

Comparison of dynamic texture classification and recognition is realized between two approaches. Generally, it is concluded that two approaches have strong ability to describe and classify dynamic textures. However, both methods have dealed with problems on some of the sets. These problems arise because the methods could not fully achieved describing those sets in terms of models and features. We have also observed that taking out these sets from test sets recognition rates increased sharply. Evaluating both methods with achieved scores on recognition rates, both methods scored similar results until including all sets into test. With full test on database, deterministic method achieved better results. However these results also can be classified as successful, they are not as good as reported scores in corresponding works in literature. When we deeply analyse, the reasons for these differences can be explained as follows. In stochastic approach, for achieving better results, method needs learning model parameters from longer sequences because the only features used are determined by models. In our study, we have created our database with 60 frames long and in terms of longer sequences this number is not yet enough. In deterministic approach, for achieving better results, method calculates features from optical flow and texture regularity. No matter on features obtained by optical flow, however some may be on texture regularity. So we have studied on improving the proposed method in literature, we have mixed up and with time constraints within we could not optimized our improvements on this measure. We believe that it is possible to increase successity of this method one step further.

Finally, we have also improved the texture regularity measure which is one of the feature extraction methods of deterministic approach. Drawbacks of proposed methods are stated and possible improvements to proposed methods in literature are defined and realized. The results are compared with the proposed methods. The results on the well known texture database are given. Calculated texture regularities which are given for this database show that our method is successful on classifying textures. However achieved success on qualitative evaluation, some faults are also reported which are caused by terms of calculation drawbacks which can be easily solved. It is shown that texture regularity measuring is nothing than interpreting the contrast curves. While we have studied on subject we have discovered and proposed improvements, but we believe that we could not yet optimized these innovations.

As a result of our analysis on dynamic texture classification, it has to be stated here that temporal information is more important than the spatial information. This is easily observable when both methods in this study and methods in literature are examined in detail. For realizing a successive dynamic texture classification, a method has to handle and focus on temporal properties. This does not mean that spatial information is redundant and can be ignored, contrary to this, spatial properties must be well defined for dynamic textures and additionally variation of these spatial properties must be defined between frames. This aspect of spatial information addresses the importance of relation between frames which are also a property of temporal domain.

As a future work, by the help of this thesis, more accurate calculations for texture regularities can be defined. Also, the classification powers of two methods discussed in this thesis can be combined.

REFERENCES

[1] G.Doretto, “Dynamic Texture Modeling”, M.S. Thesis, University of California, 2002.

[2] K.D.Cock and B.D.Moor, “Subspace angles between linear stochastic models”, Proceedings of 39th IEEE Conference on Decision and Control, pp.1561-1566, 2000.

[3] R.J.Martin, “A Metric for ARMA Processes”, IEEE Transactions On Signal Processing, vol.48, no.4, pp.1164-1170, 2000.

[4] R.Peteri and D.Chetverikov, “Dynamic texture recognition using normal flow and texture regularity”, Proc. 2nd Iberian Conference on Pattern Recognition and Image Analysis, vol.3523, pp.223-230, 2005.

[5] D.Chetverikov, “Pattern Regularity as a Visual Key”, Image and Vision Computing, 18:975-985, 2000.

[6] B.K.P.Horn and B.G.Schunck, “Determining Optical Flow”, Artificial Intelligence, vol.17, pp.185-203, 1981.

[7] R.Peteri and D.Chetverikov, “Qualitative Characterization Of Dynamic Textures For Video Retrieval”, In Proceedings ICCVG, 2004.

[8] R.Polana and R.C.Nelson, “Recognition of Motion from Temporal Texture”, IEEE Proceedings, Computer Vision and Pattern Recognition, pp.129-134, 1992.

[9] M.Szummer and R.W.Picard, “Temporal Texture Modeling”, Proc. International Conference on Image Processing, vol.3, pp.823-826, 1996.

[10] G.Doretto, D.Cremers, P.Favaro and S.Soatto, “Dynamic Texture Segmentation”, Proc. Ninth IEEE International Conference on Computer Vision, vol.2, pp.1236-1242, 2003.

[11] L.I.Smith, "A Tutorial On Principal Components Analysis", 2002. (http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf, last visited on November, 2005)

[12] M.E.Wall, A.Rechtsteiner and L.M.Rocha, "Singular Value Decomposition and Principal Component Analysis", In A Practical Approach to Microarray Data Analysis, pp.91-109, 2003.

[13] R.Chetverikov and R.Peteri, "A brief survey of dynamic texture description and recognition", Proc. of the International Conference on Computer Recognition Systems, 2005.

[14] R.Deriche, "Recursively Implementing the Gaussian and Its Derivatives", In Proc. Second International Conference On Image Processing, pp.263-267, 1992.

[15] K.B.Sookocheff, "Computing Texture Regularity", Image Processing and Computer Vision, 2004.

[16] S.Fazekas and D.Chetverikov, "Normal Versus Complete Flow In Dynamic Texture Recognition: A Comparative Study", 4th International Workshop on Texture Analysis and Synthesis, 2005.

[17] P.V.Overschee and B.De Moor, "N4SID: Subspace Algorithms for the Identification of Combined Deterministic-Stochastic Systems", Automatica, vol.30, pp.75–93, 1994.

[18] K.D.Cock and B.D.Moor, "Subspace angles and distances between ARMA models", Proceedings of the International Symposium on the Mathematical Theory of Networks and Systems, 2000.

[19] P.Saisan, G.Doretto, Y.N.Wu and S.Soatto, "Dynamic Texture Recognition", IEEE Proceedings, Computer Vision and Pattern Recognition, vol.2, pp.58-63, 2001.

[20] R.J.Martin, "Autoregression and Cepstrum-Domain Filtering", Signal Processing, vol.76, no.1, pp.93-97, 1999.

[21] A.Bissacco, A.Chiuso, Y.Ma and S. Soatto, "Recognition of Human Gaits", In Proc. of the IEEE Intl. Conf. on Comp. Vision and Patt. Recog., pp.401-417, 2001.

[22] S.Soatto and A.Chiuso, "Snippets of System Identification in Computer Vision", In Proc. of the IFAC Int. Symposium on System Identification (SYSID), 2003.

[23] D.Chetverikov and A.Hanbury, "Finding Defects in Texture Using Regularity and Local Orientation", Pattern Recognition, 35:203-218, 2002.

[24] C.H.Peh and L.F.Cheong, "Synergizing Spatial and Temporal Texture", IEEE Transactions on Image Processing, 11(10):pp.1179–1191, 2002.

[25] K.Otsuka, T.Horikoshi, S.Suzuki and M.Fujii, "Feature Extraction of Temporal Texture Based On Spatiotemporal Motion Trajectory", In Int. Conf. on Pattern Recog. ICPR'98, vol.2, pp.1047–1051, 1998.

[26] A.D.Bimbo, P.Nesi and J.L.C.Sanz, "Optical Flow Computation Using Extended Constraints", IEEE Transactions on Image Processing, vol.5, no.6, pp.701-719, 1996.

[27] S.C.Fu and P.Kovesi, "Extracting Differential Invariants of Motion Directly From Optical Flow", 13th School of Computer Science & Software Engineering Research Conference, Yanchep, Western Australia, 20th–21st September 2004.

[28] A.Mitiche, R.Grisell and J.K.Aggarwal, "On Smoothness of a Vector Field: Application to Optical Flow", PAMI(10), no.6, pp.943-949, 1988.

[29] R.Polana and R.C.Nelson, "Temporal Texture and Activity Recognition", In Motion-Based Recognition, Mubarak Shah and Ramesh Jain Editors. Kluwer Academic, 1997.

[30] G.Doretto, A.Chiuso, Y.N.Wu and S.Soatto, “Dynamic Textures”, International Journal of Computer Vision 51(2), pp.91–109, 2003.

[31] G.Doretto and S.Soatto, “Editable Dynamic Textures”, IEEE Proceedings, Computer Vision and Pattern Recognition, vol.2, pp.137-142, 2003.

[32] Y.Wang and S.C.Zhu, “Analysis and Synthesis of Textured Motion: Particles and Waves”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.26, no.10, pp.1348-1363, 2004.

[33] S.C.Zhu, C.Guo, Y.Wang and Z.Xu, “What are Textons?”, Int’l Journal on Computer Vision: Special Issue on Texture, vol.62, no.1-2, pp.121-143, 2005.

[34] R.W.Picard, T.Kabir and F.Liu, “Real-time Recognition with the entire Brodatz Texture Database”, IEEE Conf. on Computer Vision and Pattern Recognition, pp.638-639, 1993.

[35] Z.Bar-Joseph, “Statistical Learning of Multi-Dimensional Textures”, M.S. Thesis, The Hebrew University of Jurusalem, 1999.

[36] A.B.Chan and N.Vasconcelos, “Probabilistic Kernels for the Classification of Auto-Regressive Visual Processes”, IEEE Conf. on Computer Vision and Pattern Recognition, vol.1, pp.846-851, 2005.

[37] M.O.Szummer, “Temporal Texture Modeling”, M.S. Thesis, Massachusetts Institute of Technology, 1995.

[38] T.M.Bayık, “Automatic Target Recognition In Infrared Imagery”, M.S. Thesis, Middle East Technical University, 2004.

[39] M.Singh and S.Singh, “Spatial Texture Analysis: A Comparative Study”, Proc. 15th International Conference on Pattern Recognition, 2002.

[40] F.Liu, "Modeling Spatial and Temporal Textures", Ph.D. Thesis, Massachusetts Institute of Technology, 1997.

[41] Y.Z.Wang and S.C.Zhu, "Analysis and Synthesis of Textured Motion: Particle, Wave and Cartoon Sketch", IEEE Trans. on Pattern Analysis and Machine Intelligence, 2004.

[42] R.Polana and R.Nelson, "Detecting Activities", IEEE Proceedings, Computer Vision and Pattern Recognition, pp.2–7, 1993.

[43] R.Polana and R.Nelson, "Recognizing Activities", IEEE Proceedings of the 12th IAPR International Conference on Computer Vision & Image Processing, vol.1, pp.815-818, 1994.

[44] Z.Bar-Joseph, R.El-Yaniv, D.Lischinski and M.Werman, "Texture Mixing and Texture Movie Synthesis using Statistical Learning", IEEE Transactions on Visualization and Computer Graphics, vol.7, pp.120-135, 2001.

[45] V.Kwatra, A.Schödl, I.Essa, G.Turk and A.Bobick, "Graphcut Textures: Image and Video Synthesis using Graph Cuts", Proc. ACM, SIGGRAPH, pp.277-286, 2003.

[46] L.Y.Wei and M.Levoy, "Fast Texture Synthesis using Tree-structured Vector Quantization", Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, pp.479-488, 2000.

[47] J.S.De Bonet, "Multiresolution Sampling Procedure for Analysis and Synthesis of Texture Images", SIGGRAPH, pp.361-368, 1997.

[48] J.Portilla and E.P.Simoncelli, "Texture Representation and Synthesis using Correlation of Complex Wavelet Coefficient Magnitudes", CSIC Technical Report #54, 1999.

[49] A.A.Efros and T.K.Leung, "Texture Synthesis by Non-parametric Sampling", The Proceedings of the Seventh IEEE International Conference on Computer Vision, vol.2, pp.1033-1038, 1999.

[50] D.J.Heeger and J.R.Bergeny, "Pyramid-Based Texture Analysis/Synthesis", Proc. International Conference on Image Processing, vol.3, pp.648-651, 1995.

[51] K.Popat and R.W.Picard, "Novel cluster-based probability model for texture synthesis classification and compression", Visual Communications and Image Processing, pp.756-768, 1993.

[52] A.Schödl, "Multi Dimensional Exemplar-Based Texture Synthesis", Ph.D. Thesis, Georgia Institute of Technology, 2002.

[53] R.J.Schalkoff, "Pattern Recognition: Statistical, Structural and Neural Approaches", John Wiley & Sons, 1992.

[54] G.H.Golub and C.F.V.Loan, "Matrix Computations", The John Hopkins University Press, 1989.

[55] L.Ljung, "System Identification: Theory for the User", Prentice-Hall, 1987.

[56] R.Johansson, "System Modeling & Identification", Prentice-Hall, 1993.

[57] F.M.Callier and C.H.Desoer, "Linear System Theory", Springer-Verlag, 1991.

[58] Wikipedia, the free encyclopedia, http://www.wikipedia.org/, last visited on December, 2005.

[59] MIT Temporal Texture Database, http://vismod.media.mit.edu/pub/szummer/temporal-texture/raw/, last visited on November, 2005.

[60] Brodatz Texture Database, http://www.ux.his.no/~tranden/brodatz.html, last visited on November, 2005.

[61] Y.Wang, J.Ostermann and Y-Q.Zhang, "Digital Video Processing and Communications", Prentice-Hall, 2001.

[62] A.K.Jain, "Fundamentals of Digital Image Processing", Prentice-Hall, 1989.

[63] R.Wilson and M.Spann, "Image Segmentation and Uncertainty", Research Studies Press Ltd, 1988.

[64] J.R.Smith, "Integrated Spatial and Feature Image Systems: Retrieval, Analysis and Compression", Ph.D. Thesis, Columbia University, 1997.

[65] R.C.Nelson and R.Polana, "Qualitative Recognition of Motion using Temporal Texture", CVGIP: Image Understanding, vol.56, pp.78-89, 1992.

[66] P.Bouthemy and R.Fablet, "Motion Characterization from Temporal Co-occurrences of Local Motion-based Measures for Video Indexing", In Proc. Int. Conf. Pattern Recognition, vol.1, pp.905-908, 1998.

[67] R.Fablet and P.Bouthemy, "Motion Recognition using Spatio-temporal Random Walks in Sequence of 2D Motion-related Measurements", In IEEE Int. Conf. on Image Processing, ICIP'2001, pp.652-655, 2001.

[68] R.Fablet and P.Bouthemy, "Motion Recognition using Nonparametric Image Motion Models Estimated from Temporal and Multi-scale Co-occurrence Statistics", IEEE Trans. PAMI, vol.25, pp.1619-1624, 2003.

[69] C.H.Peh and L.-F.Cheong, "Exploring Video Content in Extended Spatiotemporal Textures", In Proc. 1st European Workshop on Content-Based Multimedia Indexing, pp.147-153, 1999.

[70] Z.Lu, W.Xie, J.Pei, and J.Huang, “Dynamic Texture Recognition by Spatiotemporal Multiresolution Histogram”, In Proc. IEEE Workshop on Motion and Video Computing (WACV/MOTION’05), 2005.

[71] J.Zhong and S.Scarlaroff, “Temporal Texture Recognition Model using 3D Features”, Technical Report, MIT Media Lab Perceptual Computing, 2002.

[72] R.P.Wildes and J.R.Bergen, “Qualitative Spatiotemporal Analysis using an Oriented Energy Representation”, In Proc. European Conference on Computer Vision, pp.768-784, 2000.

[73] J.R.Smith, C.-Y.Lin, and M.Naphade, “Video Texture Indexing using Spatiotemporal Wavelets”, In IEEE Int. Conf. on Image Processing, ICIP’2002, vol.2, pp.437-440, 2002.

[74] G.Doretto, E.Jones, and S.Soatto, “Spatially Homogeneous Dynamic Textures”, In Proc. European Conference on Computer Vision, vol.2, pp.591-602, 2004.

[75] L.Yuan, F.Weng, C.Liu, and H.-Y.Shum, “Synthesizing Dynamic Texture with Closed-loop Linear Dynamic System”, In Proc. European Conference on Computer Vision, vol.2, pp.603-616, 2004.

[76] K.Fujita and S.K.Nayar, “Recognition of Dynamic Textures using Impulse Responses of State Variables”, In Proc. Third International Workshop on Texture Analysis and Synthesis, pp.31-36, 2003.

[77] S.Zhu, Y.Wu and D.Mumford, “Minimax Entropy Principle and its Application to Texture Modeling”, Neural Computation, vol.9, pp.1627–1660, 1997.